

Contents

1	Clove Manual	1
1.1	License	1
1.2	About	1
1.3	Up-to-date?	2
1.4	Maturity	2
1.5	Dependencies	2
1.6	Introduction	2
1.7	First Steps	3
1.8	How To Build User Interfaces	3
1.8.1	Building Widgets	3
1.8.2	Layout And Alignment	5
1.8.2.1	Stack Layout	5
1.8.2.2	Grid Layout	5
1.8.2.3	Finetuning	6
1.9	Widget Names	7
1.9.1	Name Scopes	7
1.10	Common Widget Functionality	7
1.10.1	Widget Property System	8
1.10.2	Common Widget Properties	8
1.10.3	Common Widget Management	8
1.11	Events	8
1.11.1	Handling An Event	9
1.11.2	Triggering An Event	9

1.12 Datasources	9
1.12.1 Native JavaScript Datasources	10
1.12.2 Datasources Model	10
1.12.2.1 Implementing A Custom Datasource	11
1.12.3 Headersources	12
1.12.4 Data Bindings	13
1.12.5 Asynchronous Data Sources	13
1.13 Dialogs	13
1.13.1 Simple Dialogs	13
1.13.2 Complex Dialogs	14
1.13.3 Popups	14
1.14 Internationalization	14
1.15 Custom Widgets	15
1.15.1 Widget Properties	15
1.15.2 Layouts And Sizing	16
1.15.2.1 Using An Existing Layout Implementation	16
1.15.2.2 Provide Custom Sizing Support	16
1.15.2.3 Prevent Name Conflicts	17
1.15.3 Best Practices	17
1.16 Styling	18
1.16.1 Clove Common Styles	18
1.17 Using Widgets In Existing Webpages	19
1.18 Class Hierarchy	19
1.19 Class List	21
1.20 clove::AbstractMenu Class Reference	23
1.20.1 Detailed Description	27
1.20.2 Member Function Documentation	27
1.20.2.1 actions()	27
1.20.2.2 addStyleClass()	27
1.20.2.3 bindProperty()	28

1.20.2.4	busy()	28
1.20.2.5	childrenWidgets()	28
1.20.2.6	computeMinimalHeightForWidth()	28
1.20.2.7	computeMinimalWidth()	29
1.20.2.8	computePreferredHeightForWidth()	29
1.20.2.9	computePreferredWidth()	29
1.20.2.10	containingNameScope()	29
1.20.2.11	declareProperty()	30
1.20.2.12	doinit()	30
1.20.2.13	doinitEarly()	30
1.20.2.14	doresize()	30
1.20.2.15	doStandaloneResizing()	31
1.20.2.16	effectivelyEnabled()	31
1.20.2.17	effectiveVisibility()	31
1.20.2.18	enabled()	31
1.20.2.19	focus()	31
1.20.2.20	getMinimalHeightForWidth()	31
1.20.2.21	getMinimalWidth()	32
1.20.2.22	getPreferredHeightForWidth()	32
1.20.2.23	getPreferredWidth()	32
1.20.2.24	getProperty()	32
1.20.2.25	hasFocus()	33
1.20.2.26	horizontalStretchAffinity()	33
1.20.2.27	hstretch()	33
1.20.2.28	init()	33
1.20.2.29	innerSize()	34
1.20.2.30	isAlive()	34
1.20.2.31	isStyleClass()	34
1.20.2.32	mayFocus()	34
1.20.2.33	name()	34

1.20.2.34	nameScope()	35
1.20.2.35	OnActionTriggered()	35
1.20.2.36	OnBeforeSubactionsExpanded()	35
1.20.2.37	OnDestroyed()	35
1.20.2.38	OnKeyDown()	35
1.20.2.39	OnKeyPress()	35
1.20.2.40	OnKeyUp()	36
1.20.2.41	OnPropertyChanged()	36
1.20.2.42	OnResized()	36
1.20.2.43	OnVisibilityChanged()	36
1.20.2.44	outerSize()	36
1.20.2.45	parentWidget()	36
1.20.2.46	registerBusy()	37
1.20.2.47	relayout()	37
1.20.2.48	remove()	37
1.20.2.49	removeStyleClass()	37
1.20.2.50	resize()	38
1.20.2.51	setActions()	38
1.20.2.52	setBusy()	38
1.20.2.53	setDoStandaloneResizing()	38
1.20.2.54	setEnabled()	39
1.20.2.55	setHorizontalStretchAffinity()	39
1.20.2.56	setHstretch()	39
1.20.2.57	setMayFocus()	39
1.20.2.58	setName()	40
1.20.2.59	setProperty()	40
1.20.2.60	setStrictHorizontalSizing()	40
1.20.2.61	setStrictVerticalSizing()	41
1.20.2.62	setStyle()	41
1.20.2.63	setStyleClass()	41

1.20.2.64 setStyleClassAssigned()	41
1.20.2.65 setVerticalStretchAffinity()	42
1.20.2.66 setVisibility()	42
1.20.2.67 setVstretch()	42
1.20.2.68 strictHorizontalSizing()	43
1.20.2.69 strictVerticalSizing()	43
1.20.2.70 style()	43
1.20.2.71 styleClass()	43
1.20.2.72 unregisterBusy()	43
1.20.2.73 verticalStretchAffinity()	44
1.20.2.74 visibility()	44
1.20.2.75 vstretch()	44
1.21 clove::AjaxAsyncDatasource Class Reference	44
1.21.1 Detailed Description	46
1.21.2 Constructor & Destructor Documentation	46
1.21.2.1 AjaxAsyncDatasource()	46
1.21.3 Member Function Documentation	47
1.21.3.1 async_pull()	47
1.21.3.2 async_push()	47
1.21.3.3 changeValue()	48
1.21.3.4 clearCache()	48
1.21.3.5 columnCount()	48
1.21.3.6 columnHeaderVisible()	48
1.21.3.7 do_async_pull()	49
1.21.3.8 do_async_push()	49
1.21.3.9 getAjaxId()	49
1.21.3.10 getColumnHeader()	50
1.21.3.11 getMetadata()	50
1.21.3.12 getRowHeader()	50
1.21.3.13 getValue()	51

1.21.3.14 OnDataArrived()	51
1.21.3.15 OnDataInsert()	51
1.21.3.16 OnDataRemove()	51
1.21.3.17 OnDataUpdate()	51
1.21.3.18 OnHeaderDataInsert() [1/2]	52
1.21.3.19 OnHeaderDataInsert() [2/2]	52
1.21.3.20 OnHeaderDataRemove() [1/2]	52
1.21.3.21 OnHeaderDataRemove() [2/2]	52
1.21.3.22 OnHeaderDataUpdate() [1/2]	52
1.21.3.23 OnHeaderDataUpdate() [2/2]	52
1.21.3.24 OnHeaderVisibilityUpdated() [1/2]	53
1.21.3.25 OnHeaderVisibilityUpdated() [2/2]	53
1.21.3.26 parent()	53
1.21.3.27 rowCount()	53
1.21.3.28 rowHeadersVisible()	53
1.21.3.29 valuePointer()	54
1.21.3.30 valuePointerNavigateInDepth()	54
1.22 clove::AsyncDatasource Class Reference	54
1.22.1 Detailed Description	56
1.22.2 Constructor & Destructor Documentation	56
1.22.2.1 AsyncDatasource()	56
1.22.3 Member Function Documentation	56
1.22.3.1 async_pull()	56
1.22.3.2 async_push()	57
1.22.3.3 changeValue()	57
1.22.3.4 clearCache()	58
1.22.3.5 columnCount()	58
1.22.3.6 columnHeaderVisible()	58
1.22.3.7 do_async_pull()	58
1.22.3.8 do_async_push()	59

1.22.3.9 getColumnHeader()	59
1.22.3.10 getMetadata()	59
1.22.3.11 getRowHeader()	60
1.22.3.12 getValue()	60
1.22.3.13 OnDataInsert()	60
1.22.3.14 OnDataRemove()	60
1.22.3.15 OnDataUpdate()	61
1.22.3.16 OnHeaderDataInsert()	61
1.22.3.17 OnHeaderDataRemove()	61
1.22.3.18 OnHeaderDataUpdate()	61
1.22.3.19 OnHeaderVisibilityUpdated()	61
1.22.3.20 parent()	61
1.22.3.21 rowCount()	62
1.22.3.22 rowHeadersVisible()	62
1.22.3.23 valuePointer()	62
1.22.3.24 valuePointerNavigateInDepth()	63
1.23 clove::AudioPlayer Class Reference	63
1.23.1 Detailed Description	67
1.23.2 Member Function Documentation	67
1.23.2.1 addStyleClass()	67
1.23.2.2 autoPlay()	68
1.23.2.3 bindProperty()	68
1.23.2.4 busy()	68
1.23.2.5 canPlayType()	68
1.23.2.6 childrenWidgets()	69
1.23.2.7 computeMinimalHeightForWidth()	69
1.23.2.8 computeMinimalWidth()	69
1.23.2.9 computePreferredHeightForWidth()	69
1.23.2.10 computePreferredWidth()	70
1.23.2.11 containingNameScope()	70

1.23.2.12 <code>currentMediaPosition()</code>	70
1.23.2.13 <code>declareProperty()</code>	70
1.23.2.14 <code>doinit()</code>	70
1.23.2.15 <code>doinitEarly()</code>	71
1.23.2.16 <code>doresize()</code>	71
1.23.2.17 <code>doStandaloneResizing()</code>	71
1.23.2.18 <code>duration()</code>	71
1.23.2.19 <code>effectivelyEnabled()</code>	71
1.23.2.20 <code>effectiveVisibility()</code>	72
1.23.2.21 <code>enabled()</code>	72
1.23.2.22 <code>focus()</code>	72
1.23.2.23 <code>getMinimalHeightForWidth()</code>	72
1.23.2.24 <code>getMinimalWidth()</code>	72
1.23.2.25 <code>getPreferredHeightForWidth()</code>	73
1.23.2.26 <code>getPreferredWidth()</code>	73
1.23.2.27 <code>getProperty()</code>	73
1.23.2.28 <code>hasEnded()</code>	73
1.23.2.29 <code>hasFocus()</code>	74
1.23.2.30 <code>horizontalStretchAffinity()</code>	74
1.23.2.31 <code>hstretch()</code>	74
1.23.2.32 <code>init()</code>	74
1.23.2.33 <code>innerSize()</code>	74
1.23.2.34 <code>isAlive()</code>	75
1.23.2.35 <code>isPaused()</code>	75
1.23.2.36 <code>isStyleClass()</code>	75
1.23.2.37 <code>loop()</code>	75
1.23.2.38 <code>mayFocus()</code>	75
1.23.2.39 <code>muted()</code>	76
1.23.2.40 <code>name()</code>	76
1.23.2.41 <code>nameScope()</code>	76

1.23.2.42 nativeNode()	76
1.23.2.43 OnDestroyed()	76
1.23.2.44 OnDurationChanged()	76
1.23.2.45 OnHasEnded()	77
1.23.2.46 OnIsPaused()	77
1.23.2.47 OnIsPlaying()	77
1.23.2.48 OnKeyDown()	77
1.23.2.49 OnKeyPress()	77
1.23.2.50 OnKeyUp()	78
1.23.2.51 OnLoadingAborted()	78
1.23.2.52 OnPropertyChanged()	78
1.23.2.53 OnReadyForPlayback()	78
1.23.2.54 OnResized()	78
1.23.2.55 OnVisibilityChanged()	78
1.23.2.56 outerSize()	79
1.23.2.57 parentWidget()	79
1.23.2.58 pause()	79
1.23.2.59 play()	79
1.23.2.60 preload()	79
1.23.2.61 registerBusy()	79
1.23.2.62 relayout()	80
1.23.2.63 remove()	80
1.23.2.64 removeStyleClass()	80
1.23.2.65 resize()	80
1.23.2.66 setAutoPlay()	81
1.23.2.67 setBusy()	81
1.23.2.68 setCurrentMediaPosition()	81
1.23.2.69 setDoStandaloneResizing()	81
1.23.2.70 setEnabled()	82
1.23.2.71 setHorizontalStretchAffinity()	82

1.23.2.72 setHstretch()	82
1.23.2.73 setLoop()	83
1.23.2.74 setMayFocus()	83
1.23.2.75 setMuted()	83
1.23.2.76 setName()	83
1.23.2.77 setPreload()	84
1.23.2.78 setProperty()	84
1.23.2.79 setShowControls()	84
1.23.2.80 setSource()	85
1.23.2.81 setStrictHorizontalSizing()	85
1.23.2.82 setStrictVerticalSizing()	85
1.23.2.83 setStyle()	85
1.23.2.84 setStyleClass()	86
1.23.2.85 setStyleClassAssigned()	86
1.23.2.86 setVerticalStretchAffinity()	86
1.23.2.87 setVisibility()	87
1.23.2.88 setVolume()	87
1.23.2.89 setVstretch()	87
1.23.2.90 showControls()	87
1.23.2.91 source()	88
1.23.2.92 strictHorizontalSizing()	88
1.23.2.93 strictVerticalSizing()	88
1.23.2.94 style()	88
1.23.2.95 styleClass()	88
1.23.2.96 unregisterBusy()	88
1.23.2.97 verticalStretchAffinity()	89
1.23.2.98 visibility()	89
1.23.2.99 volume()	89
1.23.2.100vstretch()	89
1.24 clove::Border Class Reference	90

1.24.1 Detailed Description	93
1.24.2 Member Function Documentation	93
1.24.2.1 addStyleClass()	93
1.24.2.2 bindProperty()	93
1.24.2.3 body()	94
1.24.2.4 busy()	94
1.24.2.5 childrenWidgets()	94
1.24.2.6 computeMinimalHeightForWidth()	94
1.24.2.7 computeMinimalWidth()	94
1.24.2.8 computePreferredHeightForWidth()	95
1.24.2.9 computePreferredWidth()	95
1.24.2.10 containingNameScope()	95
1.24.2.11 declareProperty()	95
1.24.2.12 doinit()	96
1.24.2.13 doinitEarly()	96
1.24.2.14 doresize()	96
1.24.2.15 doStandaloneResizing()	96
1.24.2.16 effectivelyEnabled()	96
1.24.2.17 effectiveVisibility()	97
1.24.2.18 enabled()	97
1.24.2.19 focus()	97
1.24.2.20 getMinimalHeightForWidth()	97
1.24.2.21 getMinimalWidth()	97
1.24.2.22 getPreferredHeightForWidth()	98
1.24.2.23 getPreferredWidth()	98
1.24.2.24 getProperty()	98
1.24.2.25 hasFocus()	98
1.24.2.26 horizontalStretchAffinity()	99
1.24.2.27 hstretch()	99
1.24.2.28 init()	99

1.24.2.29 innerSize()	99
1.24.2.30 isAlive()	99
1.24.2.31 isStyleClass()	100
1.24.2.32 mayFocus()	100
1.24.2.33 name()	100
1.24.2.34 nameScope()	100
1.24.2.35 OnDestroyed()	100
1.24.2.36 OnKeyDown()	101
1.24.2.37 OnKeyPress()	101
1.24.2.38 OnKeyUp()	101
1.24.2.39 OnPropertyChanged()	101
1.24.2.40 OnResized()	101
1.24.2.41 OnVisibilityChanged()	101
1.24.2.42 outerSize()	102
1.24.2.43 parentWidget()	102
1.24.2.44 registerBusy()	102
1.24.2.45 relayout()	102
1.24.2.46 remove()	102
1.24.2.47 removeStyleClass()	103
1.24.2.48 resize()	103
1.24.2.49 setBody()	103
1.24.2.50 setBusy()	103
1.24.2.51 setDoStandaloneResizing()	104
1.24.2.52 setEnabled()	104
1.24.2.53 setHorizontalStretchAffinity()	104
1.24.2.54 setHstretch()	105
1.24.2.55 setMayFocus()	105
1.24.2.56 setName()	105
1.24.2.57 setProperty()	105
1.24.2.58 setStrictHorizontalSizing()	106

1.24.2.59 setStrictVerticalSizing()	106
1.24.2.60 setStyle()	106
1.24.2.61 setStyleClass()	107
1.24.2.62 setStyleClassAssigned()	107
1.24.2.63 setVerticalStretchAffinity()	107
1.24.2.64 setVisibility()	107
1.24.2.65 setVstretch()	108
1.24.2.66 strictHorizontalSizing()	108
1.24.2.67 strictVerticalSizing()	108
1.24.2.68 style()	108
1.24.2.69 styleClass()	109
1.24.2.70 unregisterBusy()	109
1.24.2.71 verticalStretchAffinity()	109
1.24.2.72 visibility()	109
1.24.2.73 vstretch()	109
1.25 clove::Button Class Reference	110
1.25.1 Detailed Description	113
1.25.2 Member Function Documentation	113
1.25.2.1 addStyleClass()	113
1.25.2.2 bindProperty()	114
1.25.2.3 busy()	114
1.25.2.4 checkable()	114
1.25.2.5 checked()	114
1.25.2.6 childrenWidgets()	114
1.25.2.7 computeMinimalHeightForWidth()	114
1.25.2.8 computeMinimalWidth()	115
1.25.2.9 computePreferredHeightForWidth()	115
1.25.2.10 computePreferredWidth()	115
1.25.2.11 containingNameScope()	115
1.25.2.12 declareProperty()	116

1.25.2.13 doinit()	116
1.25.2.14 doinitEarly()	116
1.25.2.15 doresize()	116
1.25.2.16 doStandaloneResizing()	117
1.25.2.17 effectivelyEnabled()	117
1.25.2.18 effectiveVisibility()	117
1.25.2.19 enabled()	117
1.25.2.20 focus()	117
1.25.2.21 getMinimalHeightForWidth()	117
1.25.2.22 getMinimalWidth()	118
1.25.2.23 getPreferredHeightForWidth()	118
1.25.2.24 getPreferredWidth()	118
1.25.2.25 getProperty()	118
1.25.2.26 hasFocus()	119
1.25.2.27 horizontalStretchAffinity()	119
1.25.2.28 hstretch()	119
1.25.2.29 icon()	119
1.25.2.30 init()	119
1.25.2.31 innerSize()	120
1.25.2.32 isAlive()	120
1.25.2.33 isStyleClass()	120
1.25.2.34 label()	120
1.25.2.35 mayFocus()	120
1.25.2.36 name()	121
1.25.2.37 nameScope()	121
1.25.2.38 OnClicked()	121
1.25.2.39 OnDestroyed()	121
1.25.2.40 OnKeyDown()	121
1.25.2.41 OnKeyPress()	121
1.25.2.42 OnKeyUp()	122

1.25.2.43 OnPropertyChanged()	122
1.25.2.44 OnResized()	122
1.25.2.45 OnVisibilityChanged()	122
1.25.2.46 outerSize()	122
1.25.2.47 parentWidget()	122
1.25.2.48 registerBusy()	123
1.25.2.49 relayout()	123
1.25.2.50 remove()	123
1.25.2.51 removeStyleClass()	123
1.25.2.52 resize()	124
1.25.2.53 setBusy()	124
1.25.2.54 setCheckable()	124
1.25.2.55 setChecked()	124
1.25.2.56 setDoStandaloneResizing()	125
1.25.2.57 setEnabled()	125
1.25.2.58 setHorizontalStretchAffinity()	125
1.25.2.59 setHstretch()	125
1.25.2.60 setIcon()	126
1.25.2.61 setLabel()	126
1.25.2.62 setMayFocus()	126
1.25.2.63 setName()	127
1.25.2.64 setProperty()	127
1.25.2.65 setStrictHorizontalSizing()	127
1.25.2.66 setStrictVerticalSizing()	127
1.25.2.67 setStyle()	128
1.25.2.68 setStyleClass()	128
1.25.2.69 setStyleClassAssigned()	128
1.25.2.70 setVerticalStretchAffinity()	129
1.25.2.71 setVisibility()	129
1.25.2.72 setVstretch()	129

1.25.2.73 strictHorizontalSizing()	129
1.25.2.74 strictVerticalSizing()	130
1.25.2.75 style()	130
1.25.2.76 styleClass()	130
1.25.2.77 unregisterBusy()	130
1.25.2.78 verticalStretchAffinity()	130
1.25.2.79 visibility()	131
1.25.2.80 vstretch()	131
1.26 clove::Carousel Class Reference	131
1.26.1 Detailed Description	135
1.26.2 Member Function Documentation	135
1.26.2.1 addStyleClass()	135
1.26.2.2 addTab()	136
1.26.2.3 bindProperty()	136
1.26.2.4 busy()	136
1.26.2.5 childrenWidgets()	136
1.26.2.6 computeMinimalHeightForWidth()	137
1.26.2.7 computeMinimalWidth()	137
1.26.2.8 computePreferredHeightForWidth()	137
1.26.2.9 computePreferredWidth()	137
1.26.2.10 containingNameScope()	138
1.26.2.11 currentTab()	138
1.26.2.12 declareProperty()	138
1.26.2.13 doinit()	138
1.26.2.14 doinitEarly()	139
1.26.2.15 doresize()	139
1.26.2.16 doStandaloneResizing()	139
1.26.2.17 effectivelyEnabled()	139
1.26.2.18 effectiveVisibility()	139
1.26.2.19 enabled()	140

1.26.2.20	focus()	140
1.26.2.21	getMinimalHeightForWidth()	140
1.26.2.22	getMinimalWidth()	140
1.26.2.23	getPreferredHeightForWidth()	140
1.26.2.24	getPreferredWidth()	141
1.26.2.25	getProperty()	141
1.26.2.26	hasFocus()	141
1.26.2.27	horizontalStretchAffinity()	141
1.26.2.28	hstretch()	142
1.26.2.29	init()	142
1.26.2.30	innerSize()	142
1.26.2.31	interval()	142
1.26.2.32	isAlive()	142
1.26.2.33	isPlaying()	143
1.26.2.34	isStyleClass()	143
1.26.2.35	mayFocus()	143
1.26.2.36	name()	143
1.26.2.37	nameScope()	143
1.26.2.38	next()	144
1.26.2.39	OnDestroyed()	144
1.26.2.40	OnKeyDown()	144
1.26.2.41	OnKeyPress()	144
1.26.2.42	OnKeyUp()	144
1.26.2.43	OnPropertyChanged()	144
1.26.2.44	OnResized()	145
1.26.2.45	OnTabCreationRequested()	145
1.26.2.46	OnVisibilityChanged()	145
1.26.2.47	outerSize()	145
1.26.2.48	parentWidget()	145
1.26.2.49	pause()	145

1.26.2.50 play()	146
1.26.2.51 registerBusy()	146
1.26.2.52 relayout()	146
1.26.2.53 remove()	146
1.26.2.54 removeStyleClass()	146
1.26.2.55 resize()	147
1.26.2.56 setBusy()	147
1.26.2.57 setCurrentTab()	147
1.26.2.58 setDoStandaloneResizing()	147
1.26.2.59 setEnabled()	148
1.26.2.60 setHorizontalStretchAffinity()	148
1.26.2.61 setHstretch()	148
1.26.2.62 setInterval()	149
1.26.2.63 setMayFocus()	149
1.26.2.64 setName()	149
1.26.2.65 setProperty()	149
1.26.2.66 setStrictHorizontalSizing()	150
1.26.2.67 setStrictVerticalSizing()	150
1.26.2.68 setStyle()	150
1.26.2.69 setStyleClass()	151
1.26.2.70 setStyleClassAssigned()	151
1.26.2.71 setSwitchInvisibleAnimationDuration()	151
1.26.2.72 setSwitchInvisibleAnimationName()	151
1.26.2.73 setSwitchVisibleAnimationDuration()	152
1.26.2.74 setSwitchVisibleAnimationName()	152
1.26.2.75 setTabBarLocation()	152
1.26.2.76 setTabs()	153
1.26.2.77 setUserMayAddTabs()	153
1.26.2.78 setVerticalStretchAffinity()	153
1.26.2.79 setVisibility()	153

1.26.2.80	setVstretch()	154
1.26.2.81	strictHorizontalSizing()	154
1.26.2.82	strictVerticalSizing()	154
1.26.2.83	style()	154
1.26.2.84	styleClass()	155
1.26.2.85	switchInvisibleAnimationDuration()	155
1.26.2.86	switchInvisibleAnimationName()	155
1.26.2.87	switchVisibleAnimationDuration()	155
1.26.2.88	switchVisibleAnimationName()	155
1.26.2.89	tabBarLocation()	155
1.26.2.90	tabs()	155
1.26.2.91	unregisterBusy()	155
1.26.2.92	userMayAddTabs()	156
1.26.2.93	verticalStretchAffinity()	156
1.26.2.94	visibility()	156
1.26.2.95	vstretch()	156
1.27	clove::CheckButton Class Reference	157
1.27.1	Detailed Description	160
1.27.2	Member Function Documentation	160
1.27.2.1	addStyleClass()	160
1.27.2.2	bindProperty()	160
1.27.2.3	busy()	161
1.27.2.4	checked()	161
1.27.2.5	childrenWidgets()	161
1.27.2.6	computeMinimalHeightForWidth()	161
1.27.2.7	computeMinimalWidth()	162
1.27.2.8	computePreferredHeightForWidth()	162
1.27.2.9	computePreferredWidth()	162
1.27.2.10	containingNameScope()	162
1.27.2.11	declareProperty()	162

1.27.2.12 doinit()	163
1.27.2.13 doinitEarly()	163
1.27.2.14 doresize()	163
1.27.2.15 doStandaloneResizing()	163
1.27.2.16 effectivelyEnabled()	164
1.27.2.17 effectiveVisibility()	164
1.27.2.18 enabled()	164
1.27.2.19 focus()	164
1.27.2.20 getMinimalHeightForWidth()	164
1.27.2.21 getMinimalWidth()	165
1.27.2.22 getPreferredHeightForWidth()	165
1.27.2.23 getPreferredWidth()	165
1.27.2.24 getProperty()	165
1.27.2.25 hasFocus()	166
1.27.2.26 horizontalStretchAffinity()	166
1.27.2.27 hstretch()	166
1.27.2.28 init()	166
1.27.2.29 innerSize()	166
1.27.2.30 isAlive()	167
1.27.2.31 isStyleClass()	167
1.27.2.32 label()	167
1.27.2.33 mayFocus()	167
1.27.2.34 name()	167
1.27.2.35 nameScope()	168
1.27.2.36 OnChanged()	168
1.27.2.37 OnClicked()	168
1.27.2.38 OnDestroyed()	168
1.27.2.39 OnKeyDown()	168
1.27.2.40 OnKeyPress()	168
1.27.2.41 OnKeyUp()	169

1.27.2.42 OnPropertyChanged()	169
1.27.2.43 OnResized()	169
1.27.2.44 OnVisibilityChanged()	169
1.27.2.45 outerSize()	169
1.27.2.46 parentWidget()	169
1.27.2.47 registerBusy()	170
1.27.2.48 relayout()	170
1.27.2.49 remove()	170
1.27.2.50 removeStyleClass()	170
1.27.2.51 resize()	171
1.27.2.52 setBusy()	171
1.27.2.53 setChecked()	171
1.27.2.54 setDoStandaloneResizing()	171
1.27.2.55 setEnabled()	172
1.27.2.56 setHorizontalStretchAffinity()	172
1.27.2.57 setHstretch()	172
1.27.2.58 setLabel()	172
1.27.2.59 setMayFocus()	173
1.27.2.60 setName()	173
1.27.2.61 setProperty()	173
1.27.2.62 setStrictHorizontalSizing()	174
1.27.2.63 setStrictVerticalSizing()	174
1.27.2.64 setStyle()	174
1.27.2.65 setStyleClass()	174
1.27.2.66 setStyleClassAssigned()	175
1.27.2.67 setVerticalStretchAffinity()	175
1.27.2.68 setVisibility()	175
1.27.2.69 setVstretch()	176
1.27.2.70 strictHorizontalSizing()	176
1.27.2.71 strictVerticalSizing()	176

1.27.2.72 style()	176
1.27.2.73 styleClass()	176
1.27.2.74 unregisterBusy()	176
1.27.2.75 verticalStretchAffinity()	177
1.27.2.76 visibility()	177
1.27.2.77 vstretch()	177
1.28 clove Class Reference	177
1.28.1 Detailed Description	181
1.28.2 Member Function Documentation	181
1.28.2.1 browser()	181
1.28.2.2 build()	182
1.28.2.3 conversationDialog()	182
1.28.2.4 databind()	182
1.28.2.5 getByName()	183
1.28.2.6 i18n()	183
1.28.2.7 inputDialog()	184
1.28.2.8 Invisible()	184
1.28.2.9 InvisibleCollapsed()	184
1.28.2.10 MenuSeparator()	184
1.28.2.11 messageDialog()	185
1.28.2.12 notifications()	185
1.28.2.13 populateUI()	185
1.28.2.14 rootNameScope()	186
1.28.2.15 selectionDialog()	186
1.28.2.16 Visible()	186
1.29 clove::DataBinding Class Reference	187
1.29.1 Detailed Description	187
1.29.2 Constructor & Destructor Documentation	187
1.29.2.1 DataBinding()	187
1.29.3 Member Function Documentation	187

1.29.3.1	bind()	188
1.29.3.2	unbind()	188
1.30	clove::DataBindingConverter Class Reference	188
1.30.1	Detailed Description	188
1.30.2	Member Function Documentation	188
1.30.2.1	convertFrom()	188
1.30.2.2	convertTo()	189
1.31	clove::Datasource Class Reference	189
1.31.1	Detailed Description	190
1.31.2	Member Function Documentation	190
1.31.2.1	changeValue()	190
1.31.2.2	columnCount()	190
1.31.2.3	getMetadata()	191
1.31.2.4	getValue()	191
1.31.2.5	OnDataInsert()	191
1.31.2.6	OnDataRemove()	191
1.31.2.7	OnDataUpdate()	192
1.31.2.8	parent()	192
1.31.2.9	rowCount()	192
1.31.2.10	valuePointer()	192
1.31.2.11	valuePointerNavigateInDepth()	193
1.32	clove::DatasourceValuePointer Class Reference	193
1.32.1	Detailed Description	194
1.32.2	Constructor & Destructor Documentation	194
1.32.2.1	DatasourceValuePointer()	194
1.32.3	Member Function Documentation	194
1.32.3.1	backendObject()	194
1.32.3.2	columnCount()	195
1.32.3.3	datasource()	195
1.32.3.4	equals()	195

1.32.3.5	fromPath()	195
1.32.3.6	getValue()	196
1.32.3.7	icol()	196
1.32.3.8	irow()	196
1.32.3.9	parent()	196
1.32.3.10	rowCount()	196
1.32.3.11	sibling()	196
1.32.3.12	toPath()	197
1.32.3.13	toString()	197
1.33	clove::DataView Class Reference	197
1.33.1	Detailed Description	202
1.33.2	Member Function Documentation	203
1.33.2.1	addStyleClass()	203
1.33.2.2	allowChecking()	203
1.33.2.3	allowSelection()	203
1.33.2.4	alwaysAllocateExpanderSpace()	203
1.33.2.5	bindProperty()	203
1.33.2.6	busy()	204
1.33.2.7	cellGenerator()	204
1.33.2.8	checkedCells()	204
1.33.2.9	childrenWidgets()	204
1.33.2.10	collapseCell()	204
1.33.2.11	columnsResizable()	205
1.33.2.12	computeMinimalHeightForWidth()	205
1.33.2.13	computeMinimalWidth()	205
1.33.2.14	computePreferredHeightForWidth()	205
1.33.2.15	computePreferredWidth()	206
1.33.2.16	containingNameScope()	206
1.33.2.17	datasource()	206
1.33.2.18	dataViewProcessor()	206

1.33.2.19 declareProperty()	206
1.33.2.20 doinit()	207
1.33.2.21 doinitEarly()	207
1.33.2.22 doresize()	207
1.33.2.23 doStandaloneResizing()	207
1.33.2.24 editCell()	207
1.33.2.25 editOnGesture()	208
1.33.2.26 effectivelyEnabled()	208
1.33.2.27 effectiveVisibility()	208
1.33.2.28 enabled()	208
1.33.2.29 expandCell()	208
1.33.2.30 expandCellRecursive()	209
1.33.2.31 focus()	209
1.33.2.32 getMinimalHeightForWidth()	209
1.33.2.33 getMinimalWidth()	209
1.33.2.34 getPreferredHeightForWidth()	210
1.33.2.35 getPreferredWidth()	210
1.33.2.36 getProperty()	210
1.33.2.37 granularity()	210
1.33.2.38 gridVisible()	211
1.33.2.39 hasFocus()	211
1.33.2.40 headersource()	211
1.33.2.41 hideExpanders()	211
1.33.2.42 horizontalStretchAffinity()	211
1.33.2.43 hstretch()	211
1.33.2.44 init()	211
1.33.2.45 innerSize()	212
1.33.2.46 isAlive()	212
1.33.2.47 isCellChecked()	212
1.33.2.48 isCellExpanded()	212

1.33.2.49 isCellSelected()	213
1.33.2.50 isStyleClass()	213
1.33.2.51 mayFocus()	213
1.33.2.52 name()	213
1.33.2.53 nameScope()	213
1.33.2.54 nodeActivationNeedsDoubleClick()	214
1.33.2.55 OnDestroyed()	214
1.33.2.56 OnKeyDown()	214
1.33.2.57 OnKeyPress()	214
1.33.2.58 OnKeyUp()	214
1.33.2.59 OnPropertyChanged()	214
1.33.2.60 OnResized()	215
1.33.2.61 OnSelectionChanged()	215
1.33.2.62 OnVisibilityChanged()	215
1.33.2.63 outerSize()	215
1.33.2.64 parentWidget()	215
1.33.2.65 registerBusy()	215
1.33.2.66 relayout()	216
1.33.2.67 remove()	216
1.33.2.68 removeStyleClass()	216
1.33.2.69 resize()	216
1.33.2.70 rowsResizable()	217
1.33.2.71 selectCell()	217
1.33.2.72 selection()	217
1.33.2.73 setAllowChecking()	217
1.33.2.74 setAllowSelection()	217
1.33.2.75 setAlwaysAllocateExpanderSpace()	218
1.33.2.76 setBusy()	218
1.33.2.77 setCellChecked()	218
1.33.2.78 setCellGenerator()	219

1.33.2.79	setCheckedCells()	219
1.33.2.80	setColumnsResizable()	219
1.33.2.81	setDatasource()	219
1.33.2.82	setDataViewProcessor()	220
1.33.2.83	setDoStandaloneResizing()	220
1.33.2.84	setEditOnGesture()	220
1.33.2.85	setEnabled()	221
1.33.2.86	setGranularity()	221
1.33.2.87	setGridVisible()	221
1.33.2.88	setHeadersource()	221
1.33.2.89	setHideExpanders()	222
1.33.2.90	setHorizontalStretchAffinity()	222
1.33.2.91	setHstretch()	222
1.33.2.92	setMayFocus()	223
1.33.2.93	setName()	223
1.33.2.94	setNodeActivationNeedsDoubleClick()	223
1.33.2.95	setProperty()	223
1.33.2.96	setRowsResizable()	224
1.33.2.97	setShowChangeMenu()	224
1.33.2.98	setShowOnlyFirstColumn()	224
1.33.2.99	setStrictHorizontalSizing()	225
1.33.2.100	setStrictVerticalSizing()	225
1.33.2.101	setStyle()	225
1.33.2.102	setStyleClass()	225
1.33.2.103	setStyleClassAssigned()	226
1.33.2.104	setVerticalStretchAffinity()	226
1.33.2.105	setVisibility()	226
1.33.2.106	setVstretch()	227
1.33.2.107	showChangeMenu()	227
1.33.2.108	showOnlyFirstColumn()	227

1.33.2.10	<code>strictHorizontalSizing()</code>	227
1.33.2.11	<code>strictVerticalSizing()</code>	228
1.33.2.11	<code>style()</code>	228
1.33.2.11	<code>styleClass()</code>	228
1.33.2.11	<code>unregisterBusy()</code>	228
1.33.2.11	<code>verticalStretchAffinity()</code>	228
1.33.2.11	<code>visibility()</code>	229
1.33.2.11	<code>&stretch()</code>	229
1.34	<code>clove::DateBox Class Reference</code>	229
1.34.1	Detailed Description	233
1.34.2	Member Function Documentation	233
1.34.2.1	<code>addStyleClass()</code>	233
1.34.2.2	<code>autocompletionFilter()</code>	233
1.34.2.3	<code>autocompletionItems()</code>	234
1.34.2.4	<code>autocompletionOpenForNoText()</code>	234
1.34.2.5	<code>bindProperty()</code>	234
1.34.2.6	<code>busy()</code>	234
1.34.2.7	<code>childrenWidgets()</code>	234
1.34.2.8	<code>computeMinimalHeightForWidth()</code>	235
1.34.2.9	<code>computeMinimalWidth()</code>	236
1.34.2.10	<code>computePreferredHeightForWidth()</code>	236
1.34.2.11	<code>computePreferredWidth()</code>	236
1.34.2.12	<code>containingNameScope()</code>	236
1.34.2.13	<code>date()</code>	237
1.34.2.14	<code>declareProperty()</code>	237
1.34.2.15	<code>doinit()</code>	237
1.34.2.16	<code>doinitEarly()</code>	237
1.34.2.17	<code>doresize()</code>	238
1.34.2.18	<code>doStandaloneResizing()</code>	238
1.34.2.19	<code>effectivelyEnabled()</code>	238

1.34.2.20 effectiveVisibility()	238
1.34.2.21 enabled()	238
1.34.2.22 focus()	238
1.34.2.23 getMinimalHeightForWidth()	238
1.34.2.24 getMinimalWidth()	239
1.34.2.25 getPreferredHeightForWidth()	239
1.34.2.26 getPreferredWidth()	239
1.34.2.27 getProperty()	239
1.34.2.28 hasFocus()	240
1.34.2.29 hintText()	240
1.34.2.30 horizontalStretchAffinity()	240
1.34.2.31 hstretch()	240
1.34.2.32 init()	240
1.34.2.33 innerSize()	241
1.34.2.34 isAlive()	241
1.34.2.35 isStyleClass()	241
1.34.2.36 mayFocus()	241
1.34.2.37 name()	241
1.34.2.38 nameScope()	242
1.34.2.39 OnChanged()	242
1.34.2.40 OnDestroyed()	242
1.34.2.41 OnKeyDown()	242
1.34.2.42 OnKeyPress()	242
1.34.2.43 OnKeyUp()	242
1.34.2.44 OnPopupTextSelected()	243
1.34.2.45 OnPropertyChanged()	243
1.34.2.46 OnResized()	243
1.34.2.47 OnVisibilityChanged()	243
1.34.2.48 outerSize()	243
1.34.2.49 parentWidget()	243

1.34.2.50 readOnly()	244
1.34.2.51 registerBusy()	244
1.34.2.52 relayout()	244
1.34.2.53 remove()	244
1.34.2.54 removeStyleClass()	244
1.34.2.55 resize()	245
1.34.2.56 setAutocompletionFilter()	245
1.34.2.57 setAutocompletionItems()	245
1.34.2.58 setAutocompletionOpenForNoText()	245
1.34.2.59 setBusy()	246
1.34.2.60 setDate()	246
1.34.2.61 setDoStandaloneResizing()	246
1.34.2.62 setEnabled()	247
1.34.2.63 setHintText()	247
1.34.2.64 setHorizontalStretchAffinity()	247
1.34.2.65 setHstretch()	247
1.34.2.66 setMayFocus()	248
1.34.2.67 setName()	248
1.34.2.68 setProperty()	248
1.34.2.69 setReadOnly()	249
1.34.2.70 setStrictHorizontalSizing()	249
1.34.2.71 setStrictVerticalSizing()	249
1.34.2.72 setStyle()	249
1.34.2.73 setStyleClass()	250
1.34.2.74 setStyleClassAssigned()	250
1.34.2.75 setText()	250
1.34.2.76 setTextSelection()	251
1.34.2.77 setVerticalStretchAffinity()	251
1.34.2.78 setVisibility()	251
1.34.2.79 setVstretch()	251

1.34.2.80	strictHorizontalSizing()	252
1.34.2.81	strictVerticalSizing()	252
1.34.2.82	style()	252
1.34.2.83	styleClass()	252
1.34.2.84	text()	252
1.34.2.85	textSelection()	252
1.34.2.86	unregisterBusy()	252
1.34.2.87	verticalStretchAffinity()	253
1.34.2.88	visibility()	253
1.34.2.89	vstretch()	253
1.35	clove::Dialog Class Reference	253
1.35.1	Detailed Description	257
1.35.2	Member Function Documentation	257
1.35.2.1	addStyleClass()	257
1.35.2.2	bindProperty()	257
1.35.2.3	body()	258
1.35.2.4	busy()	258
1.35.2.5	childrenWidgets()	258
1.35.2.6	close()	258
1.35.2.7	computeMinimalHeightForWidth()	258
1.35.2.8	computeMinimalWidth()	259
1.35.2.9	computePreferredHeightForWidth()	259
1.35.2.10	computePreferredWidth()	259
1.35.2.11	containingNameScope()	259
1.35.2.12	declareProperty()	259
1.35.2.13	doinit()	260
1.35.2.14	doinitEarly()	260
1.35.2.15	doresize()	260
1.35.2.16	doStandaloneResizing()	260
1.35.2.17	effectivelyEnabled()	261

1.35.2.18 effectiveVisibility()	261
1.35.2.19 enabled()	261
1.35.2.20 focus()	261
1.35.2.21 getMinimalHeightForWidth()	261
1.35.2.22 getMinimalWidth()	262
1.35.2.23 getPreferredHeightForWidth()	262
1.35.2.24 getPreferredWidth()	262
1.35.2.25 getProperty()	262
1.35.2.26 hasFocus()	263
1.35.2.27 horizontalStretchAffinity()	263
1.35.2.28 hstretch()	263
1.35.2.29 init()	263
1.35.2.30 innerSize()	263
1.35.2.31 isAlive()	264
1.35.2.32 isStyleClass()	264
1.35.2.33 mayFocus()	264
1.35.2.34 name()	264
1.35.2.35 nameScope()	264
1.35.2.36 OnDestroyed()	265
1.35.2.37 OnKeyDown()	265
1.35.2.38 OnKeyPress()	265
1.35.2.39 OnKeyUp()	265
1.35.2.40 OnPropertyChanged()	265
1.35.2.41 OnResized()	265
1.35.2.42 OnVisibilityChanged()	266
1.35.2.43 outerSize()	266
1.35.2.44 parentWidget()	266
1.35.2.45 registerBusy()	266
1.35.2.46 relayout()	266
1.35.2.47 remove()	266

1.35.2.48 removeStyleClass()	267
1.35.2.49 resize()	267
1.35.2.50 setBody()	267
1.35.2.51 setBusy()	267
1.35.2.52 setDoStandaloneResizing()	268
1.35.2.53 setEnabled()	268
1.35.2.54 setHorizontalStretchAffinity()	268
1.35.2.55 setHstretch()	269
1.35.2.56 setMayFocus()	269
1.35.2.57 setName()	269
1.35.2.58 setProperty()	269
1.35.2.59 setStrictHorizontalSizing()	270
1.35.2.60 setStrictVerticalSizing()	270
1.35.2.61 setStyle()	270
1.35.2.62 setStyleClass()	271
1.35.2.63 setStyleClassAssigned()	271
1.35.2.64 setTitle()	271
1.35.2.65 setTitleicon()	271
1.35.2.66 setVerticalStretchAffinity()	272
1.35.2.67 setVisibility()	272
1.35.2.68 setVstretch()	272
1.35.2.69 show()	273
1.35.2.70 strictHorizontalSizing()	273
1.35.2.71 strictVerticalSizing()	273
1.35.2.72 style()	273
1.35.2.73 styleClass()	273
1.35.2.74 title()	274
1.35.2.75 titleicon()	274
1.35.2.76 unregisterBusy()	274
1.35.2.77 verticalStretchAffinity()	274

1.35.2.78 visibility()	274
1.35.2.79 vstretch()	275
1.36 clove::DropDownBox Class Reference	275
1.36.1 Detailed Description	279
1.36.2 Member Function Documentation	279
1.36.2.1 addStyleClass()	279
1.36.2.2 autocompletionFilter()	279
1.36.2.3 autocompletionItems()	279
1.36.2.4 autocompletionOpenForNoText()	279
1.36.2.5 bindProperty()	280
1.36.2.6 busy()	280
1.36.2.7 childrenWidgets()	280
1.36.2.8 computeMinimalHeightForWidth()	280
1.36.2.9 computeMinimalWidth()	281
1.36.2.10 computePreferredHeightForWidth()	281
1.36.2.11 computePreferredWidth()	281
1.36.2.12 containingNameScope()	281
1.36.2.13 declareProperty()	281
1.36.2.14 doinit()	282
1.36.2.15 doinitEarly()	282
1.36.2.16 doresize()	282
1.36.2.17 doStandaloneResizing()	282
1.36.2.18 effectivelyEnabled()	283
1.36.2.19 effectiveVisibility()	283
1.36.2.20 enabled()	283
1.36.2.21 focus()	283
1.36.2.22 getMinimalHeightForWidth()	283
1.36.2.23 getMinimalWidth()	284
1.36.2.24 getPreferredHeightForWidth()	284
1.36.2.25 getPreferredWidth()	284

1.36.2.26	getProperty()	284
1.36.2.27	hasFocus()	285
1.36.2.28	hintText()	285
1.36.2.29	horizontalStretchAffinity()	285
1.36.2.30	hstretch()	285
1.36.2.31	init()	285
1.36.2.32	innerSize()	286
1.36.2.33	isAlive()	286
1.36.2.34	isStyleClass()	286
1.36.2.35	mayFocus()	286
1.36.2.36	name()	286
1.36.2.37	nameScope()	287
1.36.2.38	OnChanged()	287
1.36.2.39	OnDestroyed()	287
1.36.2.40	OnKeyDown()	287
1.36.2.41	OnKeyPress()	287
1.36.2.42	OnKeyUp()	287
1.36.2.43	OnPopupTextSelected()	288
1.36.2.44	OnPropertyChanged()	288
1.36.2.45	OnResized()	288
1.36.2.46	OnVisibilityChanged()	288
1.36.2.47	outerSize()	288
1.36.2.48	parentWidget()	288
1.36.2.49	popupItems()	289
1.36.2.50	readOnly()	289
1.36.2.51	registerBusy()	289
1.36.2.52	relayout()	289
1.36.2.53	remove()	289
1.36.2.54	removeStyleClass()	290
1.36.2.55	resize()	290

1.36.2.56 setAutocompletionFilter()	290
1.36.2.57 setAutocompletionItems()	290
1.36.2.58 setAutocompletionOpenForNoText()	291
1.36.2.59 setBusy()	291
1.36.2.60 setDoStandaloneResizing()	291
1.36.2.61 setEnabled()	292
1.36.2.62 setHintText()	292
1.36.2.63 setHorizontalStretchAffinity()	292
1.36.2.64 setHstretch()	292
1.36.2.65 setMayFocus()	293
1.36.2.66 setName()	293
1.36.2.67 setPopuItems()	293
1.36.2.68 setProperty()	294
1.36.2.69 setReadOnly()	294
1.36.2.70 setStrictHorizontalSizing()	294
1.36.2.71 setStrictVerticalSizing()	294
1.36.2.72 setStyle()	295
1.36.2.73 setStyleClass()	295
1.36.2.74 setStyleClassAssigned()	295
1.36.2.75 setText()	296
1.36.2.76 setTextSelection()	296
1.36.2.77 setVerticalStretchAffinity()	296
1.36.2.78 setVisibility()	296
1.36.2.79 setVstretch()	297
1.36.2.80 strictHorizontalSizing()	297
1.36.2.81 strictVerticalSizing()	297
1.36.2.82 style()	297
1.36.2.83 styleClass()	298
1.36.2.84 text()	298
1.36.2.85 textSelection()	298

1.36.2.86 unregisterBusy()	298
1.36.2.87 verticalStretchAffinity()	298
1.36.2.88 visibility()	299
1.36.2.89 vstretch()	299
1.37 clove::EditBox Class Reference	299
1.37.1 Detailed Description	303
1.37.2 Member Function Documentation	303
1.37.2.1 addStyleClass()	303
1.37.2.2 autocompletionFilter()	303
1.37.2.3 autocompletionItems()	303
1.37.2.4 autocompletionOpenForNoText()	304
1.37.2.5 bindProperty()	304
1.37.2.6 busy()	304
1.37.2.7 childrenWidgets()	304
1.37.2.8 computeMinimalHeightForWidth()	304
1.37.2.9 computeMinimalWidth()	305
1.37.2.10 computePreferredHeightForWidth()	305
1.37.2.11 computePreferredWidth()	305
1.37.2.12 containingNameScope()	305
1.37.2.13 declareProperty()	306
1.37.2.14 doinit()	306
1.37.2.15 doinitEarly()	306
1.37.2.16 doresize()	306
1.37.2.17 doStandaloneResizing()	307
1.37.2.18 effectivelyEnabled()	307
1.37.2.19 effectiveVisibility()	307
1.37.2.20 enabled()	307
1.37.2.21 focus()	307
1.37.2.22 getMinimalHeightForWidth()	307
1.37.2.23 getMinimalWidth()	308

1.37.2.24 <code>getPreferredHeightForWidth()</code>	308
1.37.2.25 <code>getPreferredWidth()</code>	308
1.37.2.26 <code>getProperty()</code>	308
1.37.2.27 <code>hasFocus()</code>	309
1.37.2.28 <code>hintText()</code>	309
1.37.2.29 <code>horizontalStretchAffinity()</code>	309
1.37.2.30 <code>hstretch()</code>	309
1.37.2.31 <code>init()</code>	309
1.37.2.32 <code>innerSize()</code>	310
1.37.2.33 <code>isAlive()</code>	310
1.37.2.34 <code>isStyleClass()</code>	310
1.37.2.35 <code>mayFocus()</code>	310
1.37.2.36 <code>name()</code>	310
1.37.2.37 <code>nameScope()</code>	311
1.37.2.38 <code>OnChanged()</code>	311
1.37.2.39 <code>OnDestroyed()</code>	311
1.37.2.40 <code>OnKeyDown()</code>	311
1.37.2.41 <code>OnKeyPress()</code>	311
1.37.2.42 <code>OnKeyUp()</code>	311
1.37.2.43 <code>OnPopupTextSelected()</code>	312
1.37.2.44 <code>OnPropertyChanged()</code>	312
1.37.2.45 <code>OnResized()</code>	312
1.37.2.46 <code>OnVisibilityChanged()</code>	312
1.37.2.47 <code>outerSize()</code>	312
1.37.2.48 <code>parentWidget()</code>	312
1.37.2.49 <code>readOnly()</code>	313
1.37.2.50 <code>registerBusy()</code>	313
1.37.2.51 <code>relayout()</code>	313
1.37.2.52 <code>remove()</code>	313
1.37.2.53 <code>removeStyleClass()</code>	313

1.37.2.54	resize()	314
1.37.2.55	setAutocompletionFilter()	314
1.37.2.56	setAutocompletionItems()	314
1.37.2.57	setAutocompletionOpenForNoText()	314
1.37.2.58	setBusy()	315
1.37.2.59	setDoStandaloneResizing()	315
1.37.2.60	setEnabled()	315
1.37.2.61	setHintText()	316
1.37.2.62	setHorizontalStretchAffinity()	316
1.37.2.63	setHstretch()	316
1.37.2.64	setMayFocus()	316
1.37.2.65	setName()	317
1.37.2.66	setProperty()	317
1.37.2.67	setReadOnly()	317
1.37.2.68	setStrictHorizontalSizing()	318
1.37.2.69	setStrictVerticalSizing()	318
1.37.2.70	setStyle()	318
1.37.2.71	setStyleClass()	318
1.37.2.72	setStyleClassAssigned()	319
1.37.2.73	setText()	319
1.37.2.74	setTextSelection()	319
1.37.2.75	setVerticalStretchAffinity()	320
1.37.2.76	setVisibility()	320
1.37.2.77	setVstretch()	320
1.37.2.78	strictHorizontalSizing()	320
1.37.2.79	strictVerticalSizing()	321
1.37.2.80	style()	321
1.37.2.81	styleClass()	321
1.37.2.82	text()	321
1.37.2.83	textSelection()	321

1.37.2.84 unregisterBusy()	321
1.37.2.85 verticalStretchAffinity()	322
1.37.2.86 visibility()	322
1.37.2.87 vstretch()	322
1.38 clove::EditComboBox Class Reference	322
1.38.1 Detailed Description	326
1.38.2 Member Function Documentation	326
1.38.2.1 addStyleClass()	326
1.38.2.2 autocompletionFilter()	327
1.38.2.3 autocompletionItems()	327
1.38.2.4 autocompletionOpenForNoText()	327
1.38.2.5 bindProperty()	327
1.38.2.6 busy()	327
1.38.2.7 childrenWidgets()	328
1.38.2.8 computeMinimalHeightForWidth()	328
1.38.2.9 computeMinimalWidth()	328
1.38.2.10 computePreferredHeightForWidth()	328
1.38.2.11 computePreferredWidth()	328
1.38.2.12 containingNameScope()	329
1.38.2.13 declareProperty()	329
1.38.2.14 doinit()	329
1.38.2.15 doinitEarly()	329
1.38.2.16 doresize()	330
1.38.2.17 doStandaloneResizing()	330
1.38.2.18 effectivelyEnabled()	330
1.38.2.19 effectiveVisibility()	330
1.38.2.20 enabled()	330
1.38.2.21 focus()	330
1.38.2.22 getMinimalHeightForWidth()	330
1.38.2.23 getMinimalWidth()	331

1.38.2.24 <code>getPreferredHeightForWidth()</code>	331
1.38.2.25 <code>getPreferredWidth()</code>	331
1.38.2.26 <code>getProperty()</code>	331
1.38.2.27 <code>hasFocus()</code>	332
1.38.2.28 <code>hintText()</code>	332
1.38.2.29 <code>horizontalStretchAffinity()</code>	332
1.38.2.30 <code>hstretch()</code>	332
1.38.2.31 <code>init()</code>	332
1.38.2.32 <code>innerSize()</code>	333
1.38.2.33 <code>isAlive()</code>	333
1.38.2.34 <code>isStyleClass()</code>	333
1.38.2.35 <code>mayFocus()</code>	333
1.38.2.36 <code>name()</code>	333
1.38.2.37 <code>nameScope()</code>	334
1.38.2.38 <code>OnChanged()</code>	334
1.38.2.39 <code>OnDestroyed()</code>	334
1.38.2.40 <code>OnKeyDown()</code>	334
1.38.2.41 <code>OnKeyPress()</code>	334
1.38.2.42 <code>OnKeyUp()</code>	334
1.38.2.43 <code>OnPopupTextSelected()</code>	335
1.38.2.44 <code>OnPropertyChanged()</code>	335
1.38.2.45 <code>OnResized()</code>	335
1.38.2.46 <code>OnVisibilityChanged()</code>	335
1.38.2.47 <code>outerSize()</code>	335
1.38.2.48 <code>parentWidget()</code>	335
1.38.2.49 <code>popupItems()</code>	336
1.38.2.50 <code>readOnly()</code>	336
1.38.2.51 <code>registerBusy()</code>	336
1.38.2.52 <code>relayout()</code>	336
1.38.2.53 <code>remove()</code>	336

1.38.2.54 removeStyleClass()	337
1.38.2.55 resize()	337
1.38.2.56 setAutocompletionFilter()	337
1.38.2.57 setAutocompletionItems()	337
1.38.2.58 setAutocompletionOpenForNoText()	338
1.38.2.59 setBusy()	338
1.38.2.60 setDoStandaloneResizing()	338
1.38.2.61 setEnabled()	339
1.38.2.62 setHintText()	339
1.38.2.63 setHorizontalStretchAffinity()	339
1.38.2.64 setHstretch()	339
1.38.2.65 setMayFocus()	340
1.38.2.66 setName()	340
1.38.2.67 setPopupsItems()	340
1.38.2.68 setProperty()	341
1.38.2.69 setReadOnly()	341
1.38.2.70 setStrictHorizontalSizing()	341
1.38.2.71 setStrictVerticalSizing()	341
1.38.2.72 setStyle()	342
1.38.2.73 setStyleClass()	342
1.38.2.74 setStyleClassAssigned()	342
1.38.2.75 setText()	343
1.38.2.76 setTextSelection()	343
1.38.2.77 setVerticalStretchAffinity()	343
1.38.2.78 setVisibility()	343
1.38.2.79 setVstretch()	344
1.38.2.80 strictHorizontalSizing()	344
1.38.2.81 strictVerticalSizing()	344
1.38.2.82 style()	344
1.38.2.83 styleClass()	345

1.38.2.84	text()	345
1.38.2.85	textSelection()	345
1.38.2.86	unregisterBusy()	345
1.38.2.87	verticalStretchAffinity()	345
1.38.2.88	visibility()	346
1.38.2.89	vstretch()	346
1.39	clove::Event Class Reference	346
1.39.1	Detailed Description	346
1.39.2	Constructor & Destructor Documentation	346
1.39.2.1	Event()	347
1.39.3	Member Function Documentation	347
1.39.3.1	addHandler()	347
1.39.3.2	addHandlerOnce()	347
1.39.3.3	hasHandlers()	347
1.39.3.4	removeHandler()	348
1.39.3.5	trigger()	348
1.40	clove::EventArgs Class Reference	348
1.40.1	Detailed Description	348
1.40.2	Constructor & Destructor Documentation	349
1.40.2.1	EventArgs()	349
1.40.3	Member Function Documentation	349
1.40.3.1	skipExecution()	349
1.41	clove::Expander Class Reference	349
1.41.1	Detailed Description	353
1.41.2	Member Function Documentation	353
1.41.2.1	addStyleClass()	353
1.41.2.2	bindProperty()	353
1.41.2.3	body()	354
1.41.2.4	busy()	354
1.41.2.5	childrenWidgets()	354

1.41.2.6	collapsedIcon()	354
1.41.2.7	collapsedLabel()	354
1.41.2.8	computeMinimalHeightForWidth()	354
1.41.2.9	computeMinimalWidth()	355
1.41.2.10	computePreferredHeightForWidth()	355
1.41.2.11	computePreferredWidth()	355
1.41.2.12	containingNameScope()	355
1.41.2.13	declareProperty()	356
1.41.2.14	doinit()	356
1.41.2.15	doinitEarly()	356
1.41.2.16	doresize()	356
1.41.2.17	doStandaloneResizing()	357
1.41.2.18	effectivelyEnabled()	357
1.41.2.19	effectiveVisibility()	357
1.41.2.20	enabled()	357
1.41.2.21	expandedIcon()	357
1.41.2.22	expandedLabel()	357
1.41.2.23	focus()	357
1.41.2.24	getMinimalHeightForWidth()	357
1.41.2.25	getMinimalWidth()	358
1.41.2.26	getPreferredHeightForWidth()	358
1.41.2.27	getPreferredWidth()	358
1.41.2.28	getProperty()	358
1.41.2.29	hasFocus()	359
1.41.2.30	horizontalStretchAffinity()	359
1.41.2.31	hstretch()	359
1.41.2.32	init()	359
1.41.2.33	innerSize()	360
1.41.2.34	isAlive()	360
1.41.2.35	isExpanded()	360

1.41.2.36 isStyleClass()	360
1.41.2.37 mayFocus()	360
1.41.2.38 name()	360
1.41.2.39 nameScope()	361
1.41.2.40 OnDestroyed()	361
1.41.2.41 OnKeyDown()	361
1.41.2.42 OnKeyPress()	361
1.41.2.43 OnKeyUp()	361
1.41.2.44 OnPropertyChanged()	361
1.41.2.45 OnResized()	362
1.41.2.46 OnVisibilityChanged()	362
1.41.2.47 outerSize()	362
1.41.2.48 parentWidget()	362
1.41.2.49 registerBusy()	362
1.41.2.50 relayout()	362
1.41.2.51 remove()	362
1.41.2.52 removeStyleClass()	363
1.41.2.53 resize()	363
1.41.2.54 setBody()	363
1.41.2.55 setBusy()	363
1.41.2.56 setCollapsedIcon()	364
1.41.2.57 setCollapsedLabel()	364
1.41.2.58 setDoStandaloneResizing()	364
1.41.2.59 setEnabled()	365
1.41.2.60 setExpandedIcon()	365
1.41.2.61 setExpandedLabel()	365
1.41.2.62 setHorizontalStretchAffinity()	365
1.41.2.63 setHstretch()	366
1.41.2.64 setIsExpanded()	366
1.41.2.65 setMayFocus()	366

1.41.2.66 setName()	367
1.41.2.67 setProperty()	367
1.41.2.68 setStrictHorizontalSizing()	367
1.41.2.69 setStrictVerticalSizing()	367
1.41.2.70 setStyle()	368
1.41.2.71 setStyleClass()	368
1.41.2.72 setStyleClassAssigned()	368
1.41.2.73 setVerticalStretchAffinity()	369
1.41.2.74 setVisibility()	369
1.41.2.75 setVstretch()	369
1.41.2.76 strictHorizontalSizing()	369
1.41.2.77 strictVerticalSizing()	370
1.41.2.78 style()	370
1.41.2.79 styleClass()	370
1.41.2.80 unregisterBusy()	370
1.41.2.81 verticalStretchAffinity()	370
1.41.2.82 visibility()	371
1.41.2.83 vstretch()	371
1.42 clove::FilterProxyDatasource Class Reference	371
1.42.1 Detailed Description	372
1.42.2 Constructor & Destructor Documentation	372
1.42.2.1 FilterProxyDatasource()	373
1.42.3 Member Function Documentation	374
1.42.3.1 changeValue()	374
1.42.3.2 columnCount()	374
1.42.3.3 columnHeaderVisible()	374
1.42.3.4 getColumnHeader()	375
1.42.3.5 getMetadata()	375
1.42.3.6 getRowHeader()	375
1.42.3.7 getValue()	376

1.42.3.8	nativePointerToProxyPointer()	376
1.42.3.9	OnDataInsert()	376
1.42.3.10	OnDataRemove()	376
1.42.3.11	OnDataUpdate()	377
1.42.3.12	OnHeaderDataInsert()	377
1.42.3.13	OnHeaderDataRemove()	377
1.42.3.14	OnHeaderDataUpdate()	377
1.42.3.15	OnHeaderVisibilityUpdated()	377
1.42.3.16	parent()	377
1.42.3.17	proxyPointerToNativePointer()	378
1.42.3.18	refresh()	378
1.42.3.19	rowCount()	378
1.42.3.20	rowHeadersVisible()	378
1.42.3.21	setColumnFilter()	379
1.42.3.22	setDatasource()	379
1.42.3.23	setRowFilter()	379
1.42.3.24	valuePointer()	380
1.42.3.25	valuePointerNavigateInDepth()	380
1.43	clove::FlatLayout Class Reference	380
1.43.1	Detailed Description	381
1.43.2	Member Function Documentation	381
1.43.2.1	addChild()	381
1.43.2.2	children()	382
1.43.2.3	clearChilds()	382
1.43.2.4	collapseHidden()	382
1.43.2.5	currentView()	382
1.43.2.6	setChildren()	382
1.43.2.7	setCollapseHidden()	383
1.43.2.8	setCurrentView()	383
1.43.2.9	setSwitchInvisibleAnimationDuration()	383

1.43.2.10	setSwitchInvisibleAnimationName()	384
1.43.2.11	setSwitchVisibleAnimationDuration()	384
1.43.2.12	setSwitchVisibleAnimationName()	384
1.43.2.13	switchInvisibleAnimationDuration()	384
1.43.2.14	switchInvisibleAnimationName()	385
1.43.2.15	switchVisibleAnimationDuration()	385
1.43.2.16	switchVisibleAnimationName()	385
1.44	clove::FlatView Class Reference	385
1.44.1	Detailed Description	389
1.44.2	Member Function Documentation	389
1.44.2.1	addChild()	389
1.44.2.2	addStyleClass()	390
1.44.2.3	bindProperty()	390
1.44.2.4	busy()	390
1.44.2.5	children()	390
1.44.2.6	childrenWidgets()	390
1.44.2.7	clearChilds()	391
1.44.2.8	collapseHidden()	391
1.44.2.9	computeMinimalHeightForWidth()	391
1.44.2.10	computeMinimalWidth()	391
1.44.2.11	computePreferredHeightForWidth()	391
1.44.2.12	computePreferredWidth()	392
1.44.2.13	containingNameScope()	392
1.44.2.14	currentView()	392
1.44.2.15	declareProperty()	392
1.44.2.16	doinit()	393
1.44.2.17	doinitEarly()	393
1.44.2.18	doresize()	393
1.44.2.19	doStandaloneResizing()	393
1.44.2.20	effectivelyEnabled()	393

1.44.2.21 effectiveVisibility()	394
1.44.2.22 enabled()	394
1.44.2.23 focus()	394
1.44.2.24 getMinimalHeightForWidth()	394
1.44.2.25 getMinimalWidth()	394
1.44.2.26 getPreferredHeightForWidth()	395
1.44.2.27 getPreferredWidth()	395
1.44.2.28 getProperty()	395
1.44.2.29 hasFocus()	395
1.44.2.30 horizontalStretchAffinity()	396
1.44.2.31 hstretch()	396
1.44.2.32 init()	396
1.44.2.33 innerSize()	396
1.44.2.34 isAlive()	396
1.44.2.35 isStyleClass()	397
1.44.2.36 mayFocus()	397
1.44.2.37 name()	397
1.44.2.38 nameScope()	397
1.44.2.39 OnDestroyed()	397
1.44.2.40 OnKeyDown()	398
1.44.2.41 OnKeyPress()	398
1.44.2.42 OnKeyUp()	398
1.44.2.43 OnPropertyChanged()	398
1.44.2.44 OnResized()	398
1.44.2.45 OnVisibilityChanged()	398
1.44.2.46 outerSize()	399
1.44.2.47 parentWidget()	399
1.44.2.48 registerBusy()	399
1.44.2.49 relayout()	399
1.44.2.50 remove()	399

1.44.2.51 removeStyleClass()	400
1.44.2.52 resize()	400
1.44.2.53 setBusy()	400
1.44.2.54 setChildren()	400
1.44.2.55 setCollapseHidden()	401
1.44.2.56 setCurrentView()	401
1.44.2.57 setDoStandaloneResizing()	401
1.44.2.58 setEnabled()	402
1.44.2.59 setHorizontalStretchAffinity()	402
1.44.2.60 setHstretch()	402
1.44.2.61 setMayFocus()	402
1.44.2.62 setName()	403
1.44.2.63 setProperty()	403
1.44.2.64 setStrictHorizontalSizing()	403
1.44.2.65 setStrictVerticalSizing()	404
1.44.2.66 setStyle()	404
1.44.2.67 setStyleClass()	404
1.44.2.68 setStyleClassAssigned()	404
1.44.2.69 setSwitchInvisibleAnimationDuration()	405
1.44.2.70 setSwitchInvisibleAnimationName()	405
1.44.2.71 setSwitchVisibleAnimationDuration()	405
1.44.2.72 setSwitchVisibleAnimationName()	406
1.44.2.73 setVerticalStretchAffinity()	406
1.44.2.74 setVisibility()	406
1.44.2.75 setVstretch()	406
1.44.2.76 strictHorizontalSizing()	407
1.44.2.77 strictVerticalSizing()	407
1.44.2.78 style()	407
1.44.2.79 styleClass()	407
1.44.2.80 switchInvisibleAnimationDuration()	407

1.44.2.81	switchInvisibleAnimationName()	408
1.44.2.82	switchVisibleAnimationDuration()	408
1.44.2.83	switchVisibleAnimationName()	408
1.44.2.84	unregisterBusy()	408
1.44.2.85	verticalStretchAffinity()	408
1.44.2.86	visibility()	409
1.44.2.87	vstretch()	409
1.45	clove::Form Class Reference	409
1.45.1	Detailed Description	412
1.45.2	Member Function Documentation	412
1.45.2.1	addStyleClass()	412
1.45.2.2	bindProperty()	413
1.45.2.3	busy()	413
1.45.2.4	childrenWidgets()	413
1.45.2.5	computeMinimalHeightForWidth()	413
1.45.2.6	computeMinimalWidth()	414
1.45.2.7	computePreferredHeightForWidth()	414
1.45.2.8	computePreferredWidth()	414
1.45.2.9	containingNameScope()	414
1.45.2.10	declareProperty()	414
1.45.2.11	doinit()	415
1.45.2.12	doinitEarly()	415
1.45.2.13	doresize()	415
1.45.2.14	doStandaloneResizing()	415
1.45.2.15	effectivelyEnabled()	416
1.45.2.16	effectiveVisibility()	416
1.45.2.17	enabled()	416
1.45.2.18	focus()	416
1.45.2.19	getMinimalHeightForWidth()	416
1.45.2.20	getMinimalWidth()	417

1.45.2.21 <code>getPreferredHeightForWidth()</code>	417
1.45.2.22 <code>getPreferredWidth()</code>	417
1.45.2.23 <code>getProperty()</code>	417
1.45.2.24 <code>hasFocus()</code>	418
1.45.2.25 <code>horizontalStretchAffinity()</code>	418
1.45.2.26 <code>hstretch()</code>	418
1.45.2.27 <code>init()</code>	418
1.45.2.28 <code>innerSize()</code>	418
1.45.2.29 <code>isAlive()</code>	419
1.45.2.30 <code>isStyleClass()</code>	419
1.45.2.31 <code>mayFocus()</code>	419
1.45.2.32 <code>name()</code>	419
1.45.2.33 <code>nameScope()</code>	419
1.45.2.34 <code>OnDestroyed()</code>	420
1.45.2.35 <code>OnKeyDown()</code>	420
1.45.2.36 <code>OnKeyPress()</code>	420
1.45.2.37 <code>OnKeyUp()</code>	420
1.45.2.38 <code>OnPropertyChanged()</code>	420
1.45.2.39 <code>OnResized()</code>	420
1.45.2.40 <code>OnVisibilityChanged()</code>	421
1.45.2.41 <code>outerSize()</code>	421
1.45.2.42 <code>parentWidget()</code>	421
1.45.2.43 <code>registerBusy()</code>	421
1.45.2.44 <code>relayout()</code>	421
1.45.2.45 <code>remove()</code>	421
1.45.2.46 <code>removeStyleClass()</code>	422
1.45.2.47 <code>resize()</code>	422
1.45.2.48 <code>sections()</code>	422
1.45.2.49 <code>setBusy()</code>	422
1.45.2.50 <code>setDoStandaloneResizing()</code>	423

1.45.2.51	setEnabled()	423
1.45.2.52	setHorizontalStretchAffinity()	423
1.45.2.53	setHstretch()	423
1.45.2.54	setMayFocus()	424
1.45.2.55	setName()	424
1.45.2.56	setProperty()	424
1.45.2.57	setSections()	425
1.45.2.58	setStrictHorizontalSizing()	425
1.45.2.59	setStrictVerticalSizing()	425
1.45.2.60	setStyle()	425
1.45.2.61	setStyleClass()	426
1.45.2.62	setStyleClassAssigned()	426
1.45.2.63	setVerticalStretchAffinity()	426
1.45.2.64	setVisibility()	427
1.45.2.65	setVstretch()	427
1.45.2.66	strictHorizontalSizing()	427
1.45.2.67	strictVerticalSizing()	427
1.45.2.68	style()	427
1.45.2.69	styleClass()	428
1.45.2.70	unregisterBusy()	428
1.45.2.71	verticalStretchAffinity()	428
1.45.2.72	visibility()	428
1.45.2.73	vstretch()	428
1.46	clove::Grid Class Reference	429
1.46.1	Detailed Description	432
1.46.2	Member Function Documentation	432
1.46.2.1	addChild()	432
1.46.2.2	addStyleClass()	433
1.46.2.3	bindProperty()	433
1.46.2.4	busy()	433

1.46.2.5	children()	433
1.46.2.6	childrenWidgets()	433
1.46.2.7	clearChilds()	434
1.46.2.8	computeMinimalHeightForWidth()	434
1.46.2.9	computeMinimalWidth()	434
1.46.2.10	computePreferredHeightForWidth()	434
1.46.2.11	computePreferredWidth()	434
1.46.2.12	containingNameScope()	435
1.46.2.13	declareProperty()	435
1.46.2.14	doinit()	435
1.46.2.15	doinitEarly()	435
1.46.2.16	doresize()	436
1.46.2.17	doStandaloneResizing()	436
1.46.2.18	effectivelyEnabled()	436
1.46.2.19	effectiveVisibility()	436
1.46.2.20	enabled()	436
1.46.2.21	focus()	436
1.46.2.22	getMinimalHeightForWidth()	436
1.46.2.23	getMinimalWidth()	437
1.46.2.24	getPreferredHeightForWidth()	437
1.46.2.25	getPreferredWidth()	437
1.46.2.26	getProperty()	437
1.46.2.27	hasFocus()	438
1.46.2.28	horizontalStretchAffinity()	438
1.46.2.29	hstretch()	438
1.46.2.30	init()	438
1.46.2.31	innerSize()	439
1.46.2.32	insertColumn()	439
1.46.2.33	insertRow()	439
1.46.2.34	isAlive()	439

1.46.2.35 isStyleClass()	439
1.46.2.36 mayFocus()	440
1.46.2.37 name()	440
1.46.2.38 nameScope()	440
1.46.2.39 OnDestroyed()	440
1.46.2.40 OnKeyDown()	440
1.46.2.41 OnKeyPress()	441
1.46.2.42 OnKeyUp()	441
1.46.2.43 OnPropertyChanged()	441
1.46.2.44 OnResized()	441
1.46.2.45 OnVisibilityChanged()	441
1.46.2.46 outerSize()	442
1.46.2.47 parentWidget()	442
1.46.2.48 registerBusy()	442
1.46.2.49 relayout()	442
1.46.2.50 remove()	442
1.46.2.51 removeColumn()	443
1.46.2.52 removeRow()	443
1.46.2.53 removeStyleClass()	443
1.46.2.54 resize()	444
1.46.2.55 setBusy()	444
1.46.2.56 setChildren()	444
1.46.2.57 setDoStandaloneResizing()	444
1.46.2.58 setEnabled()	445
1.46.2.59 setHorizontalStretchAffinity()	445
1.46.2.60 setHstretch()	445
1.46.2.61 setMayFocus()	445
1.46.2.62 setName()	446
1.46.2.63 setProperty()	446
1.46.2.64 setStrictHorizontalSizing()	446

1.46.2.65	setStrictVerticalSizing()	447
1.46.2.66	setStyle()	447
1.46.2.67	setStyleClass()	447
1.46.2.68	setStyleClassAssigned()	447
1.46.2.69	setVerticalStretchAffinity()	448
1.46.2.70	setVisibility()	448
1.46.2.71	setVstretch()	448
1.46.2.72	strictHorizontalSizing()	449
1.46.2.73	strictVerticalSizing()	449
1.46.2.74	style()	449
1.46.2.75	styleClass()	449
1.46.2.76	unregisterBusy()	449
1.46.2.77	verticalStretchAffinity()	450
1.46.2.78	visibility()	450
1.46.2.79	vstretch()	450
1.47	clove::GridLayout Class Reference	450
1.47.1	Detailed Description	451
1.47.2	Member Function Documentation	451
1.47.2.1	addChild()	451
1.47.2.2	children()	451
1.47.2.3	clearChilds()	452
1.47.2.4	insertColumn()	452
1.47.2.5	insertRow()	452
1.47.2.6	removeColumn()	452
1.47.2.7	removeRow()	453
1.47.2.8	setChildren()	453
1.48	clove::Headersource Class Reference	453
1.48.1	Detailed Description	454
1.48.2	Member Function Documentation	454
1.48.2.1	columnHeadersVisible()	454

1.48.2.2	getColumnHeader()	454
1.48.2.3	getRowHeader()	455
1.48.2.4	OnHeaderDataInsert()	455
1.48.2.5	OnHeaderDataRemove()	455
1.48.2.6	OnHeaderDataUpdate()	455
1.48.2.7	OnHeaderVisibilityUpdated()	455
1.48.2.8	rowHeadersVisible()	455
1.49	clove::HorizontalStack Class Reference	456
1.49.1	Detailed Description	459
1.49.2	Member Function Documentation	459
1.49.2.1	addChild()	459
1.49.2.2	addStyleClass()	460
1.49.2.3	bindProperty()	460
1.49.2.4	busy()	460
1.49.2.5	children()	460
1.49.2.6	childrenWidgets()	461
1.49.2.7	clearChilds()	461
1.49.2.8	cols()	461
1.49.2.9	computeMinimalHeightForWidth()	461
1.49.2.10	computeMinimalWidth()	461
1.49.2.11	computePreferredHeightForWidth()	462
1.49.2.12	computePreferredWidth()	462
1.49.2.13	containingNameScope()	462
1.49.2.14	declareProperty()	462
1.49.2.15	doinit()	463
1.49.2.16	doinitEarly()	463
1.49.2.17	doresize()	463
1.49.2.18	doStandaloneResizing()	463
1.49.2.19	effectivelyEnabled()	463
1.49.2.20	effectiveVisibility()	464

1.49.2.21	<code>enabled()</code>	464
1.49.2.22	<code>focus()</code>	464
1.49.2.23	<code>getMinimalHeightForWidth()</code>	464
1.49.2.24	<code>getMinimalWidth()</code>	464
1.49.2.25	<code>getPreferredHeightForWidth()</code>	465
1.49.2.26	<code>getPreferredWidth()</code>	465
1.49.2.27	<code>getProperty()</code>	465
1.49.2.28	<code>hasFocus()</code>	465
1.49.2.29	<code>horizontalStretchAffinity()</code>	466
1.49.2.30	<code>hstretch()</code>	466
1.49.2.31	<code>init()</code>	466
1.49.2.32	<code>innerSize()</code>	466
1.49.2.33	<code>insertColumn()</code>	466
1.49.2.34	<code>insertRow()</code>	467
1.49.2.35	<code>isAlive()</code>	467
1.49.2.36	<code>isStyleClass()</code>	467
1.49.2.37	<code>mayFocus()</code>	467
1.49.2.38	<code>name()</code>	468
1.49.2.39	<code>nameScope()</code>	468
1.49.2.40	<code>OnDestroyed()</code>	468
1.49.2.41	<code>OnKeyDown()</code>	468
1.49.2.42	<code>OnKeyPress()</code>	468
1.49.2.43	<code>OnKeyUp()</code>	468
1.49.2.44	<code>OnPropertyChanged()</code>	469
1.49.2.45	<code>OnResized()</code>	469
1.49.2.46	<code>OnVisibilityChanged()</code>	469
1.49.2.47	<code>outerSize()</code>	469
1.49.2.48	<code>parentWidget()</code>	469
1.49.2.49	<code>registerBusy()</code>	469
1.49.2.50	<code>relayout()</code>	470

1.49.2.51 remove()	470
1.49.2.52 removeColumn()	470
1.49.2.53 removeRow()	470
1.49.2.54 removeStyleClass()	471
1.49.2.55 resize()	471
1.49.2.56 setBusy()	471
1.49.2.57 setChildren()	471
1.49.2.58 setCols()	472
1.49.2.59 setDoStandaloneResizing()	472
1.49.2.60 setEnabled()	472
1.49.2.61 setHorizontalStretchAffinity()	473
1.49.2.62 setHstretch()	473
1.49.2.63 setMayFocus()	473
1.49.2.64 setName()	473
1.49.2.65 setProperty()	474
1.49.2.66 setStrictHorizontalSizing()	474
1.49.2.67 setStrictVerticalSizing()	474
1.49.2.68 setStyle()	475
1.49.2.69 setStyleClass()	475
1.49.2.70 setStyleClassAssigned()	475
1.49.2.71 setVerticalStretchAffinity()	475
1.49.2.72 setVisibility()	477
1.49.2.73 setVstretch()	477
1.49.2.74 strictHorizontalSizing()	477
1.49.2.75 strictVerticalSizing()	477
1.49.2.76 style()	478
1.49.2.77 styleClass()	478
1.49.2.78 unregisterBusy()	478
1.49.2.79 verticalStretchAffinity()	478
1.49.2.80 visibility()	478

1.49.2.81	vstretch()	479
1.50	clove::HtmlView Class Reference	479
1.50.1	Detailed Description	482
1.50.2	Member Function Documentation	482
1.50.2.1	addStyleClass()	482
1.50.2.2	bindProperty()	482
1.50.2.3	busy()	483
1.50.2.4	childrenWidgets()	483
1.50.2.5	computeMinimalHeightForWidth()	483
1.50.2.6	computeMinimalWidth()	483
1.50.2.7	computePreferredHeightForWidth()	484
1.50.2.8	computePreferredWidth()	484
1.50.2.9	containingNameScope()	484
1.50.2.10	contentRoot()	484
1.50.2.11	declareProperty()	484
1.50.2.12	doinit()	485
1.50.2.13	doinitEarly()	485
1.50.2.14	doresize()	485
1.50.2.15	doStandaloneResizing()	485
1.50.2.16	effectivelyEnabled()	486
1.50.2.17	effectiveVisibility()	486
1.50.2.18	enabled()	486
1.50.2.19	focus()	486
1.50.2.20	getMinimalHeightForWidth()	486
1.50.2.21	getMinimalWidth()	487
1.50.2.22	getPreferredHeightForWidth()	487
1.50.2.23	getPreferredWidth()	487
1.50.2.24	getProperty()	487
1.50.2.25	hasFocus()	488
1.50.2.26	horizontalStretchAffinity()	488

1.50.2.27 hstretch()	488
1.50.2.28 init()	488
1.50.2.29 innerSize()	488
1.50.2.30 isAlive()	489
1.50.2.31 isStyleClass()	489
1.50.2.32 mayFocus()	489
1.50.2.33 name()	489
1.50.2.34 nameScope()	489
1.50.2.35 OnDestroyed()	490
1.50.2.36 OnKeyDown()	490
1.50.2.37 OnKeyPress()	490
1.50.2.38 OnKeyUp()	490
1.50.2.39 OnPropertyChanged()	490
1.50.2.40 OnResized()	490
1.50.2.41 OnVisibilityChanged()	491
1.50.2.42 outerSize()	491
1.50.2.43 parentWidget()	491
1.50.2.44 registerBusy()	491
1.50.2.45 relayout()	491
1.50.2.46 remove()	491
1.50.2.47 removeStyleClass()	492
1.50.2.48 resize()	492
1.50.2.49 setBusy()	492
1.50.2.50 setContentRoot()	492
1.50.2.51 setDoStandaloneResizing()	493
1.50.2.52 setEnabled()	493
1.50.2.53 setHorizontalStretchAffinity()	493
1.50.2.54 setHstretch()	494
1.50.2.55 setMayFocus()	494
1.50.2.56 setName()	494

1.50.2.57	setProperty()	494
1.50.2.58	setStrictHorizontalSizing()	495
1.50.2.59	setStrictVerticalSizing()	495
1.50.2.60	setStyle()	495
1.50.2.61	setStyleClass()	496
1.50.2.62	setStyleClassAssigned()	496
1.50.2.63	setVerticalStretchAffinity()	496
1.50.2.64	setVisibility()	496
1.50.2.65	setVstretch()	497
1.50.2.66	strictHorizontalSizing()	497
1.50.2.67	strictVerticalSizing()	497
1.50.2.68	style()	497
1.50.2.69	styleClass()	498
1.50.2.70	unregisterBusy()	498
1.50.2.71	verticalStretchAffinity()	498
1.50.2.72	visibility()	498
1.50.2.73	vstretch()	498
1.51	clove::I18N Class Reference	499
1.51.1	Detailed Description	499
1.51.2	Member Function Documentation	499
1.51.2.1	addString()	499
1.51.2.2	parseLangCode()	499
1.51.2.3	setLanguage()	500
1.52	clove::Icon Class Reference	500
1.52.1	Detailed Description	500
1.52.2	Member Function Documentation	501
1.52.2.1	bySymbol()	501
1.52.2.2	byUrl()	501
1.52.2.3	createHtml()	501
1.53	clove::IconView Class Reference	502

1.53.1 Detailed Description	505
1.53.2 Member Function Documentation	505
1.53.2.1 addStyleClass()	505
1.53.2.2 bindProperty()	505
1.53.2.3 busy()	506
1.53.2.4 childrenWidgets()	506
1.53.2.5 computeMinimalHeightForWidth()	506
1.53.2.6 computeMinimalWidth()	506
1.53.2.7 computePreferredHeightForWidth()	506
1.53.2.8 computePreferredWidth()	507
1.53.2.9 containingNameScope()	507
1.53.2.10 declareProperty()	507
1.53.2.11 doinit()	507
1.53.2.12 doinitEarly()	508
1.53.2.13 doresize()	508
1.53.2.14 doStandaloneResizing()	508
1.53.2.15 effectivelyEnabled()	508
1.53.2.16 effectiveVisibility()	508
1.53.2.17 enabled()	509
1.53.2.18 focus()	509
1.53.2.19 getMinimalHeightForWidth()	509
1.53.2.20 getMinimalWidth()	509
1.53.2.21 getPreferredHeightForWidth()	509
1.53.2.22 getPreferredWidth()	510
1.53.2.23 getProperty()	510
1.53.2.24 hasFocus()	510
1.53.2.25 horizontalStretchAffinity()	510
1.53.2.26 hstretch()	511
1.53.2.27 icon()	511
1.53.2.28 init()	511

1.53.2.29 innerSize()	511
1.53.2.30 isAlive()	511
1.53.2.31 isStyleClass()	512
1.53.2.32 mayFocus()	512
1.53.2.33 name()	512
1.53.2.34 nameScope()	512
1.53.2.35 OnDestroyed()	512
1.53.2.36 OnKeyDown()	513
1.53.2.37 OnKeyPress()	513
1.53.2.38 OnKeyUp()	513
1.53.2.39 OnPropertyChanged()	513
1.53.2.40 OnResized()	513
1.53.2.41 OnVisibilityChanged()	513
1.53.2.42 outerSize()	514
1.53.2.43 parentWidget()	514
1.53.2.44 registerBusy()	514
1.53.2.45 relayout()	514
1.53.2.46 remove()	514
1.53.2.47 removeStyleClass()	515
1.53.2.48 resize()	515
1.53.2.49 setBusy()	515
1.53.2.50 setDoStandaloneResizing()	515
1.53.2.51 setEnabled()	516
1.53.2.52 setHorizontalStretchAffinity()	516
1.53.2.53 setHstretch()	516
1.53.2.54 setIcon()	517
1.53.2.55 setMayFocus()	517
1.53.2.56 setName()	517
1.53.2.57 setProperty()	517
1.53.2.58 setSize()	518

1.53.2.59	setStrictHorizontalSizing()	518
1.53.2.60	setStrictVerticalSizing()	518
1.53.2.61	setStyle()	519
1.53.2.62	setStyleClass()	519
1.53.2.63	setStyleClassAssigned()	519
1.53.2.64	setVerticalStretchAffinity()	519
1.53.2.65	setVisibility()	520
1.53.2.66	setVstretch()	520
1.53.2.67	size()	520
1.53.2.68	strictHorizontalSizing()	520
1.53.2.69	strictVerticalSizing()	521
1.53.2.70	style()	521
1.53.2.71	styleClass()	521
1.53.2.72	unregisterBusy()	521
1.53.2.73	verticalStretchAffinity()	521
1.53.2.74	visibility()	522
1.53.2.75	vstretch()	522
1.54	clove::ImageView Class Reference	522
1.54.1	Detailed Description	525
1.54.2	Member Function Documentation	525
1.54.2.1	addStyleClass()	526
1.54.2.2	bindProperty()	526
1.54.2.3	busy()	526
1.54.2.4	childrenWidgets()	526
1.54.2.5	computeMinimalHeightForWidth()	526
1.54.2.6	computeMinimalWidth()	527
1.54.2.7	computePreferredHeightForWidth()	527
1.54.2.8	computePreferredWidth()	527
1.54.2.9	containingNameScope()	527
1.54.2.10	declareProperty()	528

1.54.2.11 doinit()	528
1.54.2.12 doinitEarly()	528
1.54.2.13 doresize()	528
1.54.2.14 doStandaloneResizing()	529
1.54.2.15 effectivelyEnabled()	529
1.54.2.16 effectiveVisibility()	529
1.54.2.17 enabled()	529
1.54.2.18 focus()	529
1.54.2.19 getMinimalHeightForWidth()	529
1.54.2.20 getMinimalWidth()	530
1.54.2.21 getPreferredHeightForWidth()	530
1.54.2.22 getPreferredWidth()	530
1.54.2.23 getProperty()	530
1.54.2.24 hasFocus()	531
1.54.2.25 horizontalStretchAffinity()	531
1.54.2.26 hstretch()	531
1.54.2.27 init()	531
1.54.2.28 innerSize()	532
1.54.2.29 isAlive()	532
1.54.2.30 isStyleClass()	532
1.54.2.31 keepAspectRatio()	532
1.54.2.32 mayFocus()	532
1.54.2.33 name()	533
1.54.2.34 nameScope()	533
1.54.2.35 OnDestroyed()	533
1.54.2.36 OnKeyDown()	533
1.54.2.37 OnKeyPress()	533
1.54.2.38 OnKeyUp()	533
1.54.2.39 OnLoaded()	534
1.54.2.40 OnPropertyChanged()	534

1.54.2.41 OnResized()	534
1.54.2.42 OnVisibilityChanged()	534
1.54.2.43 outerSize()	534
1.54.2.44 parentWidget()	534
1.54.2.45 registerBusy()	535
1.54.2.46 relayout()	535
1.54.2.47 remove()	535
1.54.2.48 removeStyleClass()	535
1.54.2.49 resize()	536
1.54.2.50 setBusy()	536
1.54.2.51 setDoStandaloneResizing()	536
1.54.2.52 setEnabled()	536
1.54.2.53 setHorizontalStretchAffinity()	537
1.54.2.54 setHstretch()	537
1.54.2.55 setKeepAspectRatio()	537
1.54.2.56 setMayFocus()	537
1.54.2.57 setName()	538
1.54.2.58 setProperty()	538
1.54.2.59 setSource()	538
1.54.2.60 setStrictHorizontalSizing()	539
1.54.2.61 setStrictVerticalSizing()	539
1.54.2.62 setStyle()	539
1.54.2.63 setStyleClass()	539
1.54.2.64 setStyleClassAssigned()	540
1.54.2.65 setVerticalStretchAffinity()	540
1.54.2.66 setVisibility()	540
1.54.2.67 setVstretch()	541
1.54.2.68 setZoom()	541
1.54.2.69 source()	541
1.54.2.70 strictHorizontalSizing()	541

1.54.2.71	<code>strictVerticalSizing()</code>	541
1.54.2.72	<code>style()</code>	542
1.54.2.73	<code>styleClass()</code>	542
1.54.2.74	<code>unregisterBusy()</code>	542
1.54.2.75	<code>verticalStretchAffinity()</code>	542
1.54.2.76	<code>visibility()</code>	542
1.54.2.77	<code>vstretch()</code>	543
1.54.2.78	<code>zoom()</code>	543
1.55	<code>clove::Label</code> Class Reference	543
1.55.1	Detailed Description	546
1.55.2	Member Function Documentation	546
1.55.2.1	<code>addStyleClass()</code>	546
1.55.2.2	<code>bindProperty()</code>	547
1.55.2.3	<code>busy()</code>	547
1.55.2.4	<code>childrenWidgets()</code>	547
1.55.2.5	<code>computeMinimalHeightForWidth()</code>	547
1.55.2.6	<code>computeMinimalWidth()</code>	548
1.55.2.7	<code>computePreferredHeightForWidth()</code>	548
1.55.2.8	<code>computePreferredWidth()</code>	548
1.55.2.9	<code>containingNameScope()</code>	548
1.55.2.10	<code>declareProperty()</code>	549
1.55.2.11	<code>doinit()</code>	549
1.55.2.12	<code>doinitEarly()</code>	549
1.55.2.13	<code>doresize()</code>	549
1.55.2.14	<code>doStandaloneResizing()</code>	550
1.55.2.15	<code>effectivelyEnabled()</code>	550
1.55.2.16	<code>effectiveVisibility()</code>	550
1.55.2.17	<code>enabled()</code>	550
1.55.2.18	<code>focus()</code>	550
1.55.2.19	<code>focusBuddy()</code>	550

1.55.2.20	getMinimalHeightForWidth()	550
1.55.2.21	getMinimalWidth()	551
1.55.2.22	getPreferredHeightForWidth()	551
1.55.2.23	getPreferredWidth()	551
1.55.2.24	getProperty()	551
1.55.2.25	hasFocus()	552
1.55.2.26	horizontalStretchAffinity()	552
1.55.2.27	hstretch()	552
1.55.2.28	htmlContent()	552
1.55.2.29	init()	552
1.55.2.30	innerSize()	553
1.55.2.31	isAlive()	553
1.55.2.32	isStyleClass()	553
1.55.2.33	label()	553
1.55.2.34	mayFocus()	553
1.55.2.35	name()	554
1.55.2.36	nameScope()	554
1.55.2.37	OnDestroyed()	554
1.55.2.38	OnKeyDown()	554
1.55.2.39	OnKeyPress()	554
1.55.2.40	OnKeyUp()	554
1.55.2.41	OnPropertyChanged()	555
1.55.2.42	OnResized()	555
1.55.2.43	OnVisibilityChanged()	555
1.55.2.44	outerSize()	555
1.55.2.45	parentWidget()	555
1.55.2.46	registerBusy()	555
1.55.2.47	relayout()	556
1.55.2.48	remove()	556
1.55.2.49	removeStyleClass()	556

1.55.2.50	resize()	556
1.55.2.51	setBusy()	557
1.55.2.52	setDoStandaloneResizing()	557
1.55.2.53	setEnabled()	557
1.55.2.54	setFocusBuddy()	557
1.55.2.55	setHorizontalStretchAffinity()	558
1.55.2.56	setHstretch()	558
1.55.2.57	setHtmlContent()	558
1.55.2.58	setLabel()	559
1.55.2.59	setMayFocus()	559
1.55.2.60	setName()	559
1.55.2.61	setProperty()	559
1.55.2.62	setStrictHorizontalSizing()	560
1.55.2.63	setStrictVerticalSizing()	560
1.55.2.64	setStyle()	560
1.55.2.65	setStyleClass()	561
1.55.2.66	setStyleClassAssigned()	561
1.55.2.67	setVerticalStretchAffinity()	561
1.55.2.68	setVisibility()	561
1.55.2.69	setVstretch()	562
1.55.2.70	strictHorizontalSizing()	562
1.55.2.71	strictVerticalSizing()	562
1.55.2.72	style()	562
1.55.2.73	styleClass()	563
1.55.2.74	unregisterBusy()	563
1.55.2.75	verticalStretchAffinity()	563
1.55.2.76	visibility()	563
1.55.2.77	vstretch()	563
1.56	clove::Layout Class Reference	564
1.56.1	Detailed Description	564

1.56.2	Member Function Documentation	564
1.56.2.1	addChild()	564
1.56.2.2	children()	565
1.56.2.3	clearChilds()	565
1.56.2.4	setChildren()	565
1.57	clove::ListView Class Reference	565
1.57.1	Detailed Description	570
1.57.2	Member Function Documentation	570
1.57.2.1	addStyleClass()	570
1.57.2.2	allowChecking()	571
1.57.2.3	allowSelection()	571
1.57.2.4	alwaysAllocateExpanderSpace()	571
1.57.2.5	bindProperty()	571
1.57.2.6	busy()	572
1.57.2.7	cellGenerator()	572
1.57.2.8	checkedCells()	572
1.57.2.9	childrenWidgets()	572
1.57.2.10	collapseCell()	572
1.57.2.11	columnsResizable()	573
1.57.2.12	computeMinimalHeightForWidth()	573
1.57.2.13	computeMinimalWidth()	573
1.57.2.14	computePreferredHeightForWidth()	573
1.57.2.15	computePreferredWidth()	573
1.57.2.16	containingNameScope()	574
1.57.2.17	datasource()	574
1.57.2.18	dataViewProcessor()	574
1.57.2.19	declareProperty()	574
1.57.2.20	doinit()	575
1.57.2.21	doinitEarly()	575
1.57.2.22	doresize()	575

1.57.2.23 doStandaloneResizing()	575
1.57.2.24 editCell()	575
1.57.2.25 editOnGesture()	576
1.57.2.26 effectivelyEnabled()	576
1.57.2.27 effectiveVisibility()	576
1.57.2.28 enabled()	576
1.57.2.29 expandCell()	576
1.57.2.30 expandCellRecursive()	577
1.57.2.31 focus()	577
1.57.2.32 getMinimalHeightForWidth()	577
1.57.2.33 getMinimalWidth()	577
1.57.2.34 getPreferredHeightForWidth()	578
1.57.2.35 getPreferredWidth()	578
1.57.2.36 getProperty()	578
1.57.2.37 granularity()	578
1.57.2.38 gridVisible()	579
1.57.2.39 hasFocus()	579
1.57.2.40 headersource()	579
1.57.2.41 hideExpanders()	579
1.57.2.42 horizontalStretchAffinity()	579
1.57.2.43 hstretch()	579
1.57.2.44 init()	579
1.57.2.45 innerSize()	580
1.57.2.46 isAlive()	580
1.57.2.47 isCellChecked()	580
1.57.2.48 isCellExpanded()	580
1.57.2.49 isCellSelected()	581
1.57.2.50 isStyleClass()	581
1.57.2.51 mayFocus()	581
1.57.2.52 name()	581

1.57.2.53 <code>nameScope()</code>	581
1.57.2.54 <code>nodeActivationNeedsDoubleClick()</code>	582
1.57.2.55 <code>OnDestroyed()</code>	582
1.57.2.56 <code>OnKeyDown()</code>	582
1.57.2.57 <code>OnKeyPress()</code>	582
1.57.2.58 <code>OnKeyUp()</code>	582
1.57.2.59 <code>OnPropertyChanged()</code>	582
1.57.2.60 <code>OnResized()</code>	583
1.57.2.61 <code>OnSelectionChanged()</code>	583
1.57.2.62 <code>OnVisibilityChanged()</code>	583
1.57.2.63 <code>outerSize()</code>	583
1.57.2.64 <code>parentWidget()</code>	583
1.57.2.65 <code>registerBusy()</code>	583
1.57.2.66 <code>relayout()</code>	584
1.57.2.67 <code>remove()</code>	584
1.57.2.68 <code>removeStyleClass()</code>	584
1.57.2.69 <code>resize()</code>	584
1.57.2.70 <code>rowsResizable()</code>	585
1.57.2.71 <code>selectCell()</code>	585
1.57.2.72 <code>selection()</code>	585
1.57.2.73 <code>setAllowChecking()</code>	585
1.57.2.74 <code>setAllowSelection()</code>	585
1.57.2.75 <code>setAlwaysAllocateExpanderSpace()</code>	586
1.57.2.76 <code>setBusy()</code>	586
1.57.2.77 <code>setCellChecked()</code>	586
1.57.2.78 <code>setCellGenerator()</code>	587
1.57.2.79 <code>setCheckedCells()</code>	587
1.57.2.80 <code>setColumnsResizable()</code>	587
1.57.2.81 <code>setDatasource()</code>	587
1.57.2.82 <code>setDataViewProcessor()</code>	588

1.57.2.83 setDoStandaloneResizing()	588
1.57.2.84 setEditOnGesture()	588
1.57.2.85 setEnabled()	589
1.57.2.86 setGranularity()	589
1.57.2.87 setGridVisible()	589
1.57.2.88 setHeadersource()	589
1.57.2.89 setHideExpanders()	590
1.57.2.90 setHorizontalStretchAffinity()	590
1.57.2.91 setHstretch()	590
1.57.2.92 setMayFocus()	591
1.57.2.93 setName()	591
1.57.2.94 setNodeActivationNeedsDoubleClick()	591
1.57.2.95 setProperty()	591
1.57.2.96 setRowsResizable()	592
1.57.2.97 setShowChangeMenu()	592
1.57.2.98 setShowOnlyFirstColumn()	592
1.57.2.99 setStrictHorizontalSizing()	593
1.57.2.100setStrictVerticalSizing()	593
1.57.2.101setStyle()	593
1.57.2.102setStyleClass()	593
1.57.2.103setStyleClassAssigned()	594
1.57.2.104setVerticalStretchAffinity()	594
1.57.2.105setVisibility()	594
1.57.2.106setVstretch()	595
1.57.2.107showChangeMenu()	595
1.57.2.108showOnlyFirstColumn()	595
1.57.2.109strictHorizontalSizing()	595
1.57.2.110strictVerticalSizing()	596
1.57.2.111style()	596
1.57.2.112styleClass()	596

1.57.2.113	unregisterBusy()	596
1.57.2.114	verticalStretchAffinity()	596
1.57.2.115	visibility()	597
1.57.2.116	stretch()	597
1.58	clove::MainView Class Reference	597
1.58.1	Detailed Description	601
1.58.2	Member Function Documentation	601
1.58.2.1	actions()	601
1.58.2.2	addStyleClass()	601
1.58.2.3	bindProperty()	602
1.58.2.4	bodyLeft()	602
1.58.2.5	bodyLeftViewActionLabel()	602
1.58.2.6	bodyRight()	602
1.58.2.7	bodyRightViewActionLabel()	603
1.58.2.8	busy()	603
1.58.2.9	childrenWidgets()	603
1.58.2.10	computeMinimalHeightForWidth()	603
1.58.2.11	computeMinimalWidth()	603
1.58.2.12	computePreferredHeightForWidth()	604
1.58.2.13	computePreferredWidth()	604
1.58.2.14	containingNameScope()	604
1.58.2.15	declareProperty()	604
1.58.2.16	doinit()	605
1.58.2.17	doinitEarly()	605
1.58.2.18	doresize()	605
1.58.2.19	doStandaloneResizing()	605
1.58.2.20	effectivelyEnabled()	605
1.58.2.21	effectiveVisibility()	606
1.58.2.22	enabled()	606
1.58.2.23	focus()	606

1.58.2.24	<code>getMinimalHeightForWidth()</code>	606
1.58.2.25	<code>getMinimalWidth()</code>	606
1.58.2.26	<code>getPreferredHeightForWidth()</code>	607
1.58.2.27	<code>getPreferredWidth()</code>	607
1.58.2.28	<code>getProperty()</code>	607
1.58.2.29	<code>hasFocus()</code>	607
1.58.2.30	<code>head1()</code>	608
1.58.2.31	<code>head2()</code>	608
1.58.2.32	<code>headControl()</code>	608
1.58.2.33	<code>headControlWidget()</code>	608
1.58.2.34	<code>horizontalStretchAffinity()</code>	608
1.58.2.35	<code>hstretch()</code>	608
1.58.2.36	<code>icon()</code>	609
1.58.2.37	<code>init()</code>	609
1.58.2.38	<code>innerSize()</code>	609
1.58.2.39	<code>isAlive()</code>	609
1.58.2.40	<code>isStyleClass()</code>	609
1.58.2.41	<code>mayFocus()</code>	610
1.58.2.42	<code>name()</code>	610
1.58.2.43	<code>nameScope()</code>	610
1.58.2.44	<code>OnDestroyed()</code>	610
1.58.2.45	<code>OnKeyDown()</code>	610
1.58.2.46	<code>OnKeyPress()</code>	611
1.58.2.47	<code>OnKeyUp()</code>	611
1.58.2.48	<code>OnPropertyChanged()</code>	611
1.58.2.49	<code>OnResized()</code>	611
1.58.2.50	<code>OnVisibilityChanged()</code>	611
1.58.2.51	<code>outerSize()</code>	612
1.58.2.52	<code>parentWidget()</code>	612
1.58.2.53	<code>registerBusy()</code>	612

1.58.2.54 relayout()	612
1.58.2.55 remove()	612
1.58.2.56 removeStyleClass()	613
1.58.2.57 resize()	613
1.58.2.58 setActions()	613
1.58.2.59 setBodyLeft()	613
1.58.2.60 setBodyLeftViewActionLabel()	614
1.58.2.61 setBodyRight()	614
1.58.2.62 setBodyRightViewActionLabel()	614
1.58.2.63 setBusy()	615
1.58.2.64 setDoStandaloneResizing()	615
1.58.2.65 setEnabled()	615
1.58.2.66 setHead1()	615
1.58.2.67 setHead2()	616
1.58.2.68 setHeadControl()	616
1.58.2.69 setHorizontalStretchAffinity()	616
1.58.2.70 setHstretch()	617
1.58.2.71 setIcon()	617
1.58.2.72 setMayFocus()	617
1.58.2.73 setName()	617
1.58.2.74 setProperty()	618
1.58.2.75 setShowOnly()	618
1.58.2.76 setSidebar()	618
1.58.2.77 setSplitterPosition()	619
1.58.2.78 setStrictHorizontalSizing()	619
1.58.2.79 setStrictVerticalSizing()	619
1.58.2.80 setStyle()	619
1.58.2.81 setStyleClass()	620
1.58.2.82 setStyleClassAssigned()	620
1.58.2.83 setVerticalStretchAffinity()	620

1.58.2.84	setVisibility()	621
1.58.2.85	setVstretch()	621
1.58.2.86	showOnly()	621
1.58.2.87	sidebar()	621
1.58.2.88	splitterPosition()	621
1.58.2.89	strictHorizontalSizing()	622
1.58.2.90	strictVerticalSizing()	622
1.58.2.91	style()	622
1.58.2.92	styleClass()	622
1.58.2.93	switchToLeftBody()	622
1.58.2.94	switchToRightBody()	622
1.58.2.95	unregisterBusy()	622
1.58.2.96	verticalStretchAffinity()	623
1.58.2.97	visibility()	623
1.58.2.98	vstretch()	623
1.59	clove::MediaPlayer Class Reference	623
1.59.1	Detailed Description	628
1.59.2	Member Function Documentation	628
1.59.2.1	addStyleClass()	628
1.59.2.2	autoPlay()	628
1.59.2.3	bindProperty()	628
1.59.2.4	busy()	629
1.59.2.5	childrenWidgets()	629
1.59.2.6	computeMinimalHeightForWidth()	629
1.59.2.7	computeMinimalWidth()	629
1.59.2.8	computePreferredHeightForWidth()	629
1.59.2.9	computePreferredWidth()	630
1.59.2.10	containingNameScope()	630
1.59.2.11	currentMediaPosition()	630
1.59.2.12	declareProperty()	630

1.59.2.13 doinit()	631
1.59.2.14 doinitEarly()	631
1.59.2.15 doresize()	631
1.59.2.16 doStandaloneResizing()	631
1.59.2.17 duration()	631
1.59.2.18 effectivelyEnabled()	632
1.59.2.19 effectiveVisibility()	632
1.59.2.20 enabled()	632
1.59.2.21 focus()	632
1.59.2.22 getMinimalHeightForWidth()	632
1.59.2.23 getMinimalWidth()	633
1.59.2.24 getPreferredHeightForWidth()	633
1.59.2.25 getPreferredWidth()	633
1.59.2.26 getProperty()	633
1.59.2.27 hasEnded()	634
1.59.2.28 hasFocus()	634
1.59.2.29 horizontalStretchAffinity()	634
1.59.2.30 hstretch()	634
1.59.2.31 init()	634
1.59.2.32 innerSize()	635
1.59.2.33 isAlive()	635
1.59.2.34 isPaused()	635
1.59.2.35 isStyleClass()	635
1.59.2.36 loop()	635
1.59.2.37 mayFocus()	636
1.59.2.38 muted()	636
1.59.2.39 name()	636
1.59.2.40 nameScope()	636
1.59.2.41 nativeNode()	636
1.59.2.42 OnDestroyed()	636

1.59.2.43 OnDurationChanged()	637
1.59.2.44 OnHasEnded()	637
1.59.2.45 OnIsPaused()	637
1.59.2.46 OnIsPlaying()	637
1.59.2.47 OnKeyDown()	637
1.59.2.48 OnKeyPress()	638
1.59.2.49 OnKeyUp()	638
1.59.2.50 OnLoadingAborted()	638
1.59.2.51 OnPropertyChanged()	638
1.59.2.52 OnReadyForPlayback()	638
1.59.2.53 OnResized()	638
1.59.2.54 OnVisibilityChanged()	639
1.59.2.55 outerSize()	639
1.59.2.56 parentWidget()	639
1.59.2.57 pause()	639
1.59.2.58 play()	639
1.59.2.59 preload()	639
1.59.2.60 registerBusy()	640
1.59.2.61 relayout()	640
1.59.2.62 remove()	640
1.59.2.63 removeStyleClass()	640
1.59.2.64 resize()	641
1.59.2.65 setAutoPlay()	641
1.59.2.66 setBusy()	641
1.59.2.67 setCurrentMediaPosition()	641
1.59.2.68 setDoStandaloneResizing()	642
1.59.2.69 setEnabled()	642
1.59.2.70 setHorizontalStretchAffinity()	642
1.59.2.71 setHstretch()	642
1.59.2.72 setLoop()	643

1.59.2.73 setMayFocus()	643
1.59.2.74 setMuted()	643
1.59.2.75 setName()	644
1.59.2.76 setPreload()	644
1.59.2.77 setProperty()	644
1.59.2.78 setShowControls()	644
1.59.2.79 setSource()	645
1.59.2.80 setStrictHorizontalSizing()	645
1.59.2.81 setStrictVerticalSizing()	645
1.59.2.82 setStyle()	646
1.59.2.83 setStyleClass()	646
1.59.2.84 setStyleClassAssigned()	646
1.59.2.85 setVerticalStretchAffinity()	646
1.59.2.86 setVisibility()	647
1.59.2.87 setVolume()	647
1.59.2.88 setVstretch()	647
1.59.2.89 showControls()	648
1.59.2.90 source()	648
1.59.2.91 strictHorizontalSizing()	648
1.59.2.92 strictVerticalSizing()	648
1.59.2.93 style()	648
1.59.2.94 styleClass()	648
1.59.2.95 unregisterBusy()	648
1.59.2.96 verticalStretchAffinity()	649
1.59.2.97 visibility()	649
1.59.2.98 volume()	649
1.59.2.99 vstretch()	649
1.60 clove::Menubar Class Reference	650
1.60.1 Detailed Description	653
1.60.2 Member Function Documentation	653

1.60.2.1	actions()	653
1.60.2.2	addStyleClass()	653
1.60.2.3	bindProperty()	654
1.60.2.4	busy()	654
1.60.2.5	childrenWidgets()	654
1.60.2.6	computeMinimalHeightForWidth()	654
1.60.2.7	computeMinimalWidth()	655
1.60.2.8	computePreferredHeightForWidth()	655
1.60.2.9	computePreferredWidth()	655
1.60.2.10	containingNameScope()	655
1.60.2.11	declareProperty()	656
1.60.2.12	doinit()	656
1.60.2.13	doinitEarly()	656
1.60.2.14	doresize()	656
1.60.2.15	doStandaloneResizing()	657
1.60.2.16	effectivelyEnabled()	657
1.60.2.17	effectiveVisibility()	657
1.60.2.18	enabled()	657
1.60.2.19	focus()	657
1.60.2.20	getMinimalHeightForWidth()	657
1.60.2.21	getMinimalWidth()	658
1.60.2.22	getPreferredHeightForWidth()	658
1.60.2.23	getPreferredWidth()	658
1.60.2.24	getProperty()	658
1.60.2.25	hasFocus()	659
1.60.2.26	horizontalStretchAffinity()	659
1.60.2.27	hstretch()	659
1.60.2.28	init()	659
1.60.2.29	innerSize()	660
1.60.2.30	isAlive()	660

1.60.2.31 isStyleClass()	660
1.60.2.32 mayFocus()	660
1.60.2.33 name()	660
1.60.2.34 nameScope()	661
1.60.2.35 OnActionTriggered()	661
1.60.2.36 OnBeforeSubactionsExpanded()	661
1.60.2.37 OnDestroyed()	661
1.60.2.38 OnKeyDown()	661
1.60.2.39 OnKeyPress()	661
1.60.2.40 OnKeyUp()	662
1.60.2.41 OnPropertyChanged()	662
1.60.2.42 OnResized()	662
1.60.2.43 OnVisibilityChanged()	662
1.60.2.44 outerSize()	662
1.60.2.45 parentWidget()	662
1.60.2.46 registerBusy()	663
1.60.2.47 relayout()	663
1.60.2.48 remove()	663
1.60.2.49 removeStyleClass()	663
1.60.2.50 resize()	664
1.60.2.51 setActions()	664
1.60.2.52 setBusy()	664
1.60.2.53 setDoStandaloneResizing()	664
1.60.2.54 setEnabled()	665
1.60.2.55 setHorizontalStretchAffinity()	665
1.60.2.56 setHstretch()	665
1.60.2.57 setMayFocus()	665
1.60.2.58 setName()	666
1.60.2.59 setProperty()	666
1.60.2.60 setStrictHorizontalSizing()	666

1.60.2.61	setStrictVerticalSizing()	667
1.60.2.62	setStyle()	667
1.60.2.63	setStyleClass()	667
1.60.2.64	setStyleClassAssigned()	667
1.60.2.65	setVerticalStretchAffinity()	668
1.60.2.66	setVisibility()	668
1.60.2.67	setVstretch()	668
1.60.2.68	strictHorizontalSizing()	669
1.60.2.69	strictVerticalSizing()	669
1.60.2.70	style()	669
1.60.2.71	styleClass()	669
1.60.2.72	unregisterBusy()	669
1.60.2.73	verticalStretchAffinity()	670
1.60.2.74	visibility()	670
1.60.2.75	vstretch()	670
1.61	clove::ModalPanel Class Reference	670
1.61.1	Detailed Description	674
1.61.2	Member Function Documentation	674
1.61.2.1	addStyleClass()	674
1.61.2.2	bindProperty()	674
1.61.2.3	busy()	674
1.61.2.4	childrenWidgets()	675
1.61.2.5	computeMinimalHeightForWidth()	675
1.61.2.6	computeMinimalWidth()	675
1.61.2.7	computePreferredHeightForWidth()	675
1.61.2.8	computePreferredWidth()	675
1.61.2.9	containingNameScope()	676
1.61.2.10	declareProperty()	676
1.61.2.11	doinit()	676
1.61.2.12	doinitEarly()	676

1.61.2.13 doresize()	677
1.61.2.14 doStandaloneResizing()	677
1.61.2.15 effectivelyEnabled()	677
1.61.2.16 effectiveVisibility()	677
1.61.2.17 enabled()	677
1.61.2.18 focus()	677
1.61.2.19 fullyTransparent()	678
1.61.2.20 getMinimalHeightForWidth()	678
1.61.2.21 getMinimalWidth()	679
1.61.2.22 getPreferredHeightForWidth()	679
1.61.2.23 getPreferredWidth()	679
1.61.2.24 getProperty()	679
1.61.2.25 hasFocus()	680
1.61.2.26 horizontalStretchAffinity()	680
1.61.2.27 hstretch()	680
1.61.2.28 init()	680
1.61.2.29 innerSize()	681
1.61.2.30 isAlive()	681
1.61.2.31 isStyleClass()	681
1.61.2.32 mayFocus()	681
1.61.2.33 name()	681
1.61.2.34 nameScope()	682
1.61.2.35 OnClicked()	682
1.61.2.36 OnDestroyed()	682
1.61.2.37 OnKeyDown()	682
1.61.2.38 OnKeyPress()	682
1.61.2.39 OnKeyUp()	682
1.61.2.40 OnPropertyChanged()	683
1.61.2.41 OnResized()	683
1.61.2.42 OnVisibilityChanged()	683

1.61.2.43	outerSize()	683
1.61.2.44	parentWidget()	683
1.61.2.45	registerBusy()	683
1.61.2.46	relayout()	684
1.61.2.47	remove()	684
1.61.2.48	removeStyleClass()	684
1.61.2.49	resize()	684
1.61.2.50	setBusy()	685
1.61.2.51	setDoStandaloneResizing()	685
1.61.2.52	setEnabled()	685
1.61.2.53	setFullyTransparent()	685
1.61.2.54	setHorizontalStretchAffinity()	686
1.61.2.55	setHstretch()	686
1.61.2.56	setMayFocus()	686
1.61.2.57	setName()	687
1.61.2.58	setProperty()	687
1.61.2.59	setStrictHorizontalSizing()	687
1.61.2.60	setStrictVerticalSizing()	687
1.61.2.61	setStyle()	688
1.61.2.62	setStyleClass()	688
1.61.2.63	setStyleClassAssigned()	688
1.61.2.64	setVerticalStretchAffinity()	689
1.61.2.65	setVisibility()	689
1.61.2.66	setVstretch()	689
1.61.2.67	strictHorizontalSizing()	689
1.61.2.68	strictVerticalSizing()	690
1.61.2.69	style()	690
1.61.2.70	styleClass()	690
1.61.2.71	unregisterBusy()	690
1.61.2.72	verticalStretchAffinity()	690

1.61.2.73 visibility()	691
1.61.2.74 vstretch()	691
1.62 clove::MultilineEditBox Class Reference	691
1.62.1 Detailed Description	695
1.62.2 Member Function Documentation	695
1.62.2.1 addStyleClass()	695
1.62.2.2 autocompletionFilter()	695
1.62.2.3 autocompletionItems()	695
1.62.2.4 autocompletionOpenForNoText()	696
1.62.2.5 bindProperty()	696
1.62.2.6 busy()	696
1.62.2.7 childrenWidgets()	696
1.62.2.8 computeMinimalHeightForWidth()	696
1.62.2.9 computeMinimalWidth()	697
1.62.2.10 computePreferredHeightForWidth()	697
1.62.2.11 computePreferredWidth()	697
1.62.2.12 containingNameScope()	697
1.62.2.13 declareProperty()	698
1.62.2.14 doinit()	698
1.62.2.15 doinitEarly()	698
1.62.2.16 doresize()	698
1.62.2.17 doStandaloneResizing()	699
1.62.2.18 effectivelyEnabled()	699
1.62.2.19 effectiveVisibility()	699
1.62.2.20 enabled()	699
1.62.2.21 focus()	699
1.62.2.22 getMinimalHeightForWidth()	699
1.62.2.23 getMinimalWidth()	700
1.62.2.24 getPreferredHeightForWidth()	700
1.62.2.25 getPreferredWidth()	700

1.62.2.26	getProperty()	700
1.62.2.27	hasFocus()	701
1.62.2.28	hintText()	701
1.62.2.29	horizontalStretchAffinity()	701
1.62.2.30	hstretch()	701
1.62.2.31	init()	701
1.62.2.32	innerSize()	702
1.62.2.33	isAlive()	702
1.62.2.34	isStyleClass()	702
1.62.2.35	mayFocus()	702
1.62.2.36	name()	702
1.62.2.37	nameScope()	703
1.62.2.38	OnChanged()	703
1.62.2.39	OnDestroyed()	703
1.62.2.40	OnKeyDown()	703
1.62.2.41	OnKeyPress()	703
1.62.2.42	OnKeyUp()	703
1.62.2.43	OnPopupTextSelected()	704
1.62.2.44	OnPropertyChanged()	704
1.62.2.45	OnResized()	704
1.62.2.46	OnVisibilityChanged()	704
1.62.2.47	outerSize()	704
1.62.2.48	parentWidget()	704
1.62.2.49	readOnly()	705
1.62.2.50	registerBusy()	705
1.62.2.51	relayout()	705
1.62.2.52	remove()	705
1.62.2.53	removeStyleClass()	705
1.62.2.54	resize()	706
1.62.2.55	setAutocompletionFilter()	706

1.62.2.56 setAutocompletionItems()	706
1.62.2.57 setAutocompletionOpenForNoText()	706
1.62.2.58 setBusy()	707
1.62.2.59 setDoStandaloneResizing()	707
1.62.2.60 setEnabled()	707
1.62.2.61 setHintText()	708
1.62.2.62 setHorizontalStretchAffinity()	708
1.62.2.63 setHstretch()	708
1.62.2.64 setMayFocus()	708
1.62.2.65 setName()	709
1.62.2.66 setProperty()	709
1.62.2.67 setReadOnly()	709
1.62.2.68 setStrictHorizontalSizing()	710
1.62.2.69 setStrictVerticalSizing()	710
1.62.2.70 setStyle()	710
1.62.2.71 setStyleClass()	710
1.62.2.72 setStyleClassAssigned()	711
1.62.2.73 setText()	711
1.62.2.74 setTextSelection()	711
1.62.2.75 setVerticalStretchAffinity()	712
1.62.2.76 setVisibility()	712
1.62.2.77 setVstretch()	712
1.62.2.78 strictHorizontalSizing()	712
1.62.2.79 strictVerticalSizing()	713
1.62.2.80 style()	713
1.62.2.81 styleClass()	713
1.62.2.82 text()	713
1.62.2.83 textSelection()	713
1.62.2.84 unregisterBusy()	713
1.62.2.85 verticalStretchAffinity()	714

1.62.2.86 visibility()	714
1.62.2.87 vstretch()	714
1.63 clove::NameScope Class Reference	714
1.63.1 Detailed Description	715
1.63.2 Member Function Documentation	715
1.63.2.1 addChildNameScope()	715
1.63.2.2 getByName()	715
1.64 clove::NativeDatasource Class Reference	716
1.64.1 Detailed Description	717
1.64.2 Constructor & Destructor Documentation	717
1.64.2.1 NativeDatasource()	717
1.64.3 Member Function Documentation	718
1.64.3.1 appendColumn()	718
1.64.3.2 appendRow()	718
1.64.3.3 changeValue()	718
1.64.3.4 columnCount()	719
1.64.3.5 columnHeaderVisible()	719
1.64.3.6 getColumnHeader()	719
1.64.3.7 getMetadata()	720
1.64.3.8 getRowHeader()	720
1.64.3.9 getValue()	720
1.64.3.10 insertColumn()	720
1.64.3.11 insertRow()	721
1.64.3.12 OnDataInsert()	721
1.64.3.13 OnDataRemove()	721
1.64.3.14 OnDataUpdate()	721
1.64.3.15 OnHeaderDataInsert()	722
1.64.3.16 OnHeaderDataRemove()	722
1.64.3.17 OnHeaderDataUpdate()	722
1.64.3.18 OnHeaderVisibilityUpdated()	722

1.64.3.19	parent()	722
1.64.3.20	removeColumn()	722
1.64.3.21	removeRow()	723
1.64.3.22	root()	723
1.64.3.23	rowCount()	723
1.64.3.24	rowHeadersVisible()	724
1.64.3.25	setColumnHeader()	724
1.64.3.26	setMetadata()	724
1.64.3.27	setRootValue()	725
1.64.3.28	setRowHeader()	725
1.64.3.29	setValue()	725
1.64.3.30	valuePointer()	726
1.64.3.31	valuePointerNavigateInDepth()	726
1.64.3.32	valuePointerToNativeNode()	726
1.65	clove::NativeDatasourceNode Class Reference	727
1.65.1	Detailed Description	727
1.65.2	Member Function Documentation	727
1.65.2.1	addColumn()	728
1.65.2.2	addRow()	729
1.65.2.3	columnCount()	729
1.65.2.4	getChild()	729
1.65.2.5	getMetadata()	729
1.65.2.6	getValue()	730
1.65.2.7	insertColumn()	730
1.65.2.8	insertRow()	730
1.65.2.9	removeColumn()	730
1.65.2.10	removeRow()	731
1.65.2.11	rowCount()	731
1.65.2.12	setMetadata()	731
1.65.2.13	setValue()	731

1.65.2.14 toColumn()	732
1.65.2.15 toRow()	732
1.65.2.16 valuePointer()	732
1.66 clove::NotificationController Class Reference	732
1.66.1 Detailed Description	733
1.66.2 Member Function Documentation	733
1.66.2.1 notify()	733
1.67 clove::NumericEditBox Class Reference	733
1.67.1 Detailed Description	738
1.67.2 Member Function Documentation	738
1.67.2.1 addStyleClass()	738
1.67.2.2 autocompletionFilter()	738
1.67.2.3 autocompletionItems()	738
1.67.2.4 autocompletionOpenForNoText()	738
1.67.2.5 bindProperty()	738
1.67.2.6 busy()	739
1.67.2.7 childrenWidgets()	739
1.67.2.8 computeMinimalHeightForWidth()	739
1.67.2.9 computeMinimalWidth()	739
1.67.2.10 computePreferredHeightForWidth()	740
1.67.2.11 computePreferredWidth()	740
1.67.2.12 containingNameScope()	740
1.67.2.13 declareProperty()	740
1.67.2.14 doinit()	741
1.67.2.15 doinitEarly()	741
1.67.2.16 doresize()	741
1.67.2.17 doStandaloneResizing()	741
1.67.2.18 effectivelyEnabled()	741
1.67.2.19 effectiveVisibility()	742
1.67.2.20 enabled()	742

1.67.2.21 focus()	742
1.67.2.22 getMinimalHeightForWidth()	742
1.67.2.23 getMinimalWidth()	742
1.67.2.24 getPreferredHeightForWidth()	743
1.67.2.25 getPreferredWidth()	743
1.67.2.26 getProperty()	743
1.67.2.27 hasFocus()	743
1.67.2.28 hintText()	744
1.67.2.29 horizontalStretchAffinity()	744
1.67.2.30 hstretch()	744
1.67.2.31 init()	744
1.67.2.32 innerSize()	744
1.67.2.33 isAlive()	745
1.67.2.34 isStyleClass()	745
1.67.2.35 max()	745
1.67.2.36 mayFocus()	745
1.67.2.37 min()	745
1.67.2.38 name()	746
1.67.2.39 nameScope()	746
1.67.2.40 OnChanged()	746
1.67.2.41 OnDestroyed()	746
1.67.2.42 OnKeyDown()	746
1.67.2.43 OnKeyPress()	746
1.67.2.44 OnKeyUp()	747
1.67.2.45 OnPopupTextSelected()	747
1.67.2.46 OnPropertyChanged()	747
1.67.2.47 OnResized()	747
1.67.2.48 OnVisibilityChanged()	747
1.67.2.49 outerSize()	747
1.67.2.50 parentWidget()	748

1.67.2.51	readOnly()	748
1.67.2.52	registerBusy()	748
1.67.2.53	relayout()	748
1.67.2.54	remove()	748
1.67.2.55	removeStyleClass()	749
1.67.2.56	resize()	749
1.67.2.57	setAutocompletionFilter()	749
1.67.2.58	setAutocompletionItems()	749
1.67.2.59	setAutocompletionOpenForNoText()	750
1.67.2.60	setBusy()	750
1.67.2.61	setDoStandaloneResizing()	750
1.67.2.62	setEnabled()	751
1.67.2.63	setHintText()	751
1.67.2.64	setHorizontalStretchAffinity()	751
1.67.2.65	setHstretch()	751
1.67.2.66	setMax()	752
1.67.2.67	setMayFocus()	752
1.67.2.68	setMin()	752
1.67.2.69	setName()	753
1.67.2.70	setProperty()	753
1.67.2.71	setReadOnly()	753
1.67.2.72	setStepsize()	753
1.67.2.73	setStrictHorizontalSizing()	754
1.67.2.74	setStrictVerticalSizing()	754
1.67.2.75	setStyle()	754
1.67.2.76	setStyleClass()	755
1.67.2.77	setStyleClassAssigned()	755
1.67.2.78	setText()	755
1.67.2.79	setTextSelection()	755
1.67.2.80	setValue()	756

1.67.2.81 setVerticalStretchAffinity()	756
1.67.2.82 setVisibility()	756
1.67.2.83 setVstretch()	757
1.67.2.84 stepsize()	757
1.67.2.85 strictHorizontalSizing()	757
1.67.2.86 strictVerticalSizing()	757
1.67.2.87 style()	757
1.67.2.88 styleClass()	758
1.67.2.89 text()	758
1.67.2.90 textSelection()	758
1.67.2.91 unregisterBusy()	758
1.67.2.92 value()	758
1.67.2.93 verticalStretchAffinity()	759
1.67.2.94 visibility()	759
1.67.2.95 vstretch()	759
1.68 clove::PasswordEditBox Class Reference	759
1.68.1 Detailed Description	763
1.68.2 Member Function Documentation	763
1.68.2.1 addStyleClass()	763
1.68.2.2 autocompletionFilter()	764
1.68.2.3 autocompletionItems()	764
1.68.2.4 autocompletionOpenForNoText()	764
1.68.2.5 bindProperty()	764
1.68.2.6 busy()	764
1.68.2.7 childrenWidgets()	765
1.68.2.8 computeMinimalHeightForWidth()	765
1.68.2.9 computeMinimalWidth()	765
1.68.2.10 computePreferredHeightForWidth()	765
1.68.2.11 computePreferredWidth()	765
1.68.2.12 containingNameScope()	766

1.68.2.13 declareProperty()	766
1.68.2.14 doinit()	766
1.68.2.15 doinitEarly()	766
1.68.2.16 doresize()	767
1.68.2.17 doStandaloneResizing()	767
1.68.2.18 effectivelyEnabled()	767
1.68.2.19 effectiveVisibility()	767
1.68.2.20 enabled()	767
1.68.2.21 focus()	767
1.68.2.22 getMinimalHeightForWidth()	767
1.68.2.23 getMinimalWidth()	768
1.68.2.24 getPreferredHeightForWidth()	768
1.68.2.25 getPreferredWidth()	768
1.68.2.26 getProperty()	768
1.68.2.27 hasFocus()	769
1.68.2.28 hintText()	769
1.68.2.29 horizontalStretchAffinity()	769
1.68.2.30 hstretch()	769
1.68.2.31 init()	769
1.68.2.32 innerSize()	770
1.68.2.33 isAlive()	770
1.68.2.34 isStyleClass()	770
1.68.2.35 mayFocus()	770
1.68.2.36 name()	770
1.68.2.37 nameScope()	771
1.68.2.38 OnChanged()	771
1.68.2.39 OnDestroyed()	771
1.68.2.40 OnKeyDown()	771
1.68.2.41 OnKeyPress()	771
1.68.2.42 OnKeyUp()	771

1.68.2.43 OnPopupTextSelected()	772
1.68.2.44 OnPropertyChanged()	772
1.68.2.45 OnResized()	772
1.68.2.46 OnVisibilityChanged()	772
1.68.2.47 outerSize()	772
1.68.2.48 parentWidget()	772
1.68.2.49 readOnly()	773
1.68.2.50 registerBusy()	773
1.68.2.51 relayout()	773
1.68.2.52 remove()	773
1.68.2.53 removeStyleClass()	773
1.68.2.54 resize()	774
1.68.2.55 setAutocompletionFilter()	774
1.68.2.56 setAutocompletionItems()	774
1.68.2.57 setAutocompletionOpenForNoText()	774
1.68.2.58 setBusy()	775
1.68.2.59 setDoStandaloneResizing()	775
1.68.2.60 setEnabled()	775
1.68.2.61 setHintText()	776
1.68.2.62 setHorizontalStretchAffinity()	776
1.68.2.63 setHstretch()	776
1.68.2.64 setMayFocus()	776
1.68.2.65 setName()	777
1.68.2.66 setProperty()	777
1.68.2.67 setReadOnly()	777
1.68.2.68 setStrictHorizontalSizing()	778
1.68.2.69 setStrictVerticalSizing()	778
1.68.2.70 setStyle()	778
1.68.2.71 setStyleClass()	778
1.68.2.72 setStyleClassAssigned()	779

1.68.2.73	setText()	779
1.68.2.74	setTextSelection()	779
1.68.2.75	setVerticalStretchAffinity()	780
1.68.2.76	setVisibility()	780
1.68.2.77	setVstretch()	780
1.68.2.78	strictHorizontalSizing()	780
1.68.2.79	strictVerticalSizing()	781
1.68.2.80	style()	781
1.68.2.81	styleClass()	781
1.68.2.82	text()	781
1.68.2.83	textSelection()	781
1.68.2.84	unregisterBusy()	781
1.68.2.85	verticalStretchAffinity()	782
1.68.2.86	visibility()	782
1.68.2.87	vstretch()	782
1.69	clove::PopupMenu Class Reference	782
1.69.1	Detailed Description	786
1.69.2	Member Function Documentation	786
1.69.2.1	actions()	786
1.69.2.2	addStyleClass()	786
1.69.2.3	bindProperty()	787
1.69.2.4	busy()	787
1.69.2.5	childrenWidgets()	787
1.69.2.6	computeMinimalHeightForWidth()	787
1.69.2.7	computeMinimalWidth()	788
1.69.2.8	computePreferredHeightForWidth()	788
1.69.2.9	computePreferredWidth()	788
1.69.2.10	containingNameScope()	788
1.69.2.11	declareProperty()	789
1.69.2.12	doinit()	789

1.69.2.13 doinitEarly()	789
1.69.2.14 doresize()	789
1.69.2.15 doStandaloneResizing()	790
1.69.2.16 effectivelyEnabled()	790
1.69.2.17 effectiveVisibility()	790
1.69.2.18 enabled()	790
1.69.2.19 expanderIndicatorDirection()	790
1.69.2.20 focus()	790
1.69.2.21 getMinimalHeightForWidth()	790
1.69.2.22 getMinimalWidth()	791
1.69.2.23 getPreferredHeightForWidth()	791
1.69.2.24 getPreferredWidth()	791
1.69.2.25 getProperty()	791
1.69.2.26 hasFocus()	792
1.69.2.27 horizontalStretchAffinity()	792
1.69.2.28 hstretch()	792
1.69.2.29 init()	792
1.69.2.30 innerSize()	793
1.69.2.31 isAlive()	793
1.69.2.32 isStyleClass()	793
1.69.2.33 mayFocus()	793
1.69.2.34 name()	793
1.69.2.35 nameScope()	794
1.69.2.36 OnActionTriggered()	794
1.69.2.37 OnBeforeSubactionsExpanded()	794
1.69.2.38 OnDestroyed()	794
1.69.2.39 OnKeyDown()	794
1.69.2.40 OnKeyPress()	794
1.69.2.41 OnKeyUp()	795
1.69.2.42 OnPropertyChanged()	795

1.69.2.43 OnResized()	795
1.69.2.44 OnVisibilityChanged()	795
1.69.2.45 outerSize()	795
1.69.2.46 parentWidget()	795
1.69.2.47 registerBusy()	796
1.69.2.48 relayout()	796
1.69.2.49 remove()	796
1.69.2.50 removeStyleClass()	796
1.69.2.51 resize()	797
1.69.2.52 setActions()	797
1.69.2.53 setBusy()	797
1.69.2.54 setDoStandaloneResizing()	797
1.69.2.55 setEnabled()	798
1.69.2.56 setExpanderIndicatorDirection()	798
1.69.2.57 setHorizontalStretchAffinity()	798
1.69.2.58 setHstretch()	798
1.69.2.59 setMayFocus()	799
1.69.2.60 setName()	799
1.69.2.61 setProperty()	799
1.69.2.62 setStrictHorizontalSizing()	800
1.69.2.63 setStrictVerticalSizing()	800
1.69.2.64 setStyle()	800
1.69.2.65 setStyleClass()	800
1.69.2.66 setStyleClassAssigned()	801
1.69.2.67 setVerticalStretchAffinity()	801
1.69.2.68 setVisibility()	801
1.69.2.69 setVstretch()	802
1.69.2.70 show()	802
1.69.2.71 strictHorizontalSizing()	802
1.69.2.72 strictVerticalSizing()	802

1.69.2.73 style()	803
1.69.2.74 styleClass()	803
1.69.2.75 unregisterBusy()	803
1.69.2.76 verticalStretchAffinity()	803
1.69.2.77 visibility()	803
1.69.2.78 vstretch()	804
1.70 clove::PopupMenuButton Class Reference	804
1.70.1 Detailed Description	807
1.70.2 Member Function Documentation	807
1.70.2.1 actions()	808
1.70.2.2 addStyleClass()	808
1.70.2.3 bindProperty()	808
1.70.2.4 busy()	808
1.70.2.5 checkable()	808
1.70.2.6 checked()	809
1.70.2.7 childrenWidgets()	809
1.70.2.8 computeMinimalHeightForWidth()	809
1.70.2.9 computeMinimalWidth()	809
1.70.2.10 computePreferredHeightForWidth()	809
1.70.2.11 computePreferredWidth()	810
1.70.2.12 containingNameScope()	810
1.70.2.13 declareProperty()	810
1.70.2.14 doinit()	810
1.70.2.15 doinitEarly()	811
1.70.2.16 doresize()	811
1.70.2.17 doStandaloneResizing()	811
1.70.2.18 effectivelyEnabled()	811
1.70.2.19 effectiveVisibility()	811
1.70.2.20 enabled()	812
1.70.2.21 focus()	812

1.70.2.22	getMinimalHeightForWidth()	812
1.70.2.23	getMinimalWidth()	812
1.70.2.24	getPreferredHeightForWidth()	812
1.70.2.25	getPreferredWidth()	813
1.70.2.26	getProperty()	813
1.70.2.27	hasFocus()	813
1.70.2.28	horizontalStretchAffinity()	813
1.70.2.29	hstretch()	814
1.70.2.30	icon()	814
1.70.2.31	init()	814
1.70.2.32	innerSize()	814
1.70.2.33	isAlive()	814
1.70.2.34	isStyleClass()	815
1.70.2.35	label()	815
1.70.2.36	mayFocus()	815
1.70.2.37	name()	815
1.70.2.38	nameScope()	815
1.70.2.39	OnActionTriggered()	816
1.70.2.40	OnClicked()	816
1.70.2.41	OnDestroyed()	816
1.70.2.42	OnKeyDown()	816
1.70.2.43	OnKeyPress()	816
1.70.2.44	OnKeyUp()	816
1.70.2.45	OnPropertyChanged()	817
1.70.2.46	OnResized()	817
1.70.2.47	OnVisibilityChanged()	817
1.70.2.48	outerSize()	817
1.70.2.49	parentWidget()	817
1.70.2.50	registerBusy()	817
1.70.2.51	relayout()	818

1.70.2.52 remove()	818
1.70.2.53 removeStyleClass()	818
1.70.2.54 resize()	818
1.70.2.55 setActions()	819
1.70.2.56 setBusy()	819
1.70.2.57 setCheckable()	819
1.70.2.58 setChecked()	819
1.70.2.59 setDoStandaloneResizing()	820
1.70.2.60 setEnabled()	820
1.70.2.61 setHorizontalStretchAffinity()	820
1.70.2.62 setHstretch()	821
1.70.2.63 setIcon()	821
1.70.2.64 setLabel()	821
1.70.2.65 setMayFocus()	821
1.70.2.66 setName()	822
1.70.2.67 setProperty()	822
1.70.2.68 setStrictHorizontalSizing()	822
1.70.2.69 setStrictVerticalSizing()	823
1.70.2.70 setStyle()	823
1.70.2.71 setStyleClass()	823
1.70.2.72 setStyleClassAssigned()	823
1.70.2.73 setVerticalStretchAffinity()	824
1.70.2.74 setVisibility()	824
1.70.2.75 setVstretch()	824
1.70.2.76 strictHorizontalSizing()	825
1.70.2.77 strictVerticalSizing()	825
1.70.2.78 style()	825
1.70.2.79 styleClass()	825
1.70.2.80 unregisterBusy()	825
1.70.2.81 verticalStretchAffinity()	826

1.70.2.82	visibility()	826
1.70.2.83	vstretch()	826
1.71	clove::ProgressBar Class Reference	826
1.71.1	Detailed Description	830
1.71.2	Member Function Documentation	830
1.71.2.1	addStyleClass()	830
1.71.2.2	bindProperty()	830
1.71.2.3	busy()	831
1.71.2.4	childrenWidgets()	831
1.71.2.5	computeMinimalHeightForWidth()	831
1.71.2.6	computeMinimalWidth()	831
1.71.2.7	computePreferredHeightForWidth()	831
1.71.2.8	computePreferredWidth()	832
1.71.2.9	containingNameScope()	832
1.71.2.10	declareProperty()	832
1.71.2.11	doinit()	832
1.71.2.12	doinitEarly()	833
1.71.2.13	doresize()	833
1.71.2.14	doStandaloneResizing()	833
1.71.2.15	effectivelyEnabled()	833
1.71.2.16	effectiveVisibility()	833
1.71.2.17	enabled()	834
1.71.2.18	focus()	834
1.71.2.19	getMinimalHeightForWidth()	834
1.71.2.20	getMinimalWidth()	834
1.71.2.21	getPreferredHeightForWidth()	834
1.71.2.22	getPreferredWidth()	835
1.71.2.23	getProperty()	835
1.71.2.24	hasFocus()	835
1.71.2.25	horizontalStretchAffinity()	835

1.71.2.26 hstretch()	836
1.71.2.27 init()	836
1.71.2.28 innerSize()	836
1.71.2.29 isAlive()	836
1.71.2.30 isStyleClass()	836
1.71.2.31 label()	837
1.71.2.32 labelFunction()	837
1.71.2.33 mayFocus()	837
1.71.2.34 name()	837
1.71.2.35 nameScope()	837
1.71.2.36 OnDestroyed()	838
1.71.2.37 OnKeyDown()	838
1.71.2.38 OnKeyPress()	838
1.71.2.39 OnKeyUp()	838
1.71.2.40 OnPropertyChanged()	838
1.71.2.41 OnResized()	838
1.71.2.42 OnVisibilityChanged()	839
1.71.2.43 orientation()	839
1.71.2.44 outerSize()	839
1.71.2.45 parentWidget()	839
1.71.2.46 registerBusy()	839
1.71.2.47 relayout()	839
1.71.2.48 remove()	839
1.71.2.49 removeStyleClass()	840
1.71.2.50 resize()	840
1.71.2.51 setBusy()	840
1.71.2.52 setDoStandaloneResizing()	840
1.71.2.53 setEnabled()	841
1.71.2.54 setHorizontalStretchAffinity()	841
1.71.2.55 setHstretch()	841

1.71.2.56	setLabel()	842
1.71.2.57	setLabelFunction()	842
1.71.2.58	setMayFocus()	842
1.71.2.59	setName()	842
1.71.2.60	setOrientation()	843
1.71.2.61	setProperty()	843
1.71.2.62	setStrictHorizontalSizing()	843
1.71.2.63	setStrictVerticalSizing()	844
1.71.2.64	setStyle()	844
1.71.2.65	setStyleClass()	844
1.71.2.66	setStyleClassAssigned()	844
1.71.2.67	setValue()	845
1.71.2.68	setVerticalStretchAffinity()	845
1.71.2.69	setVisibility()	845
1.71.2.70	setVstretch()	846
1.71.2.71	strictHorizontalSizing()	846
1.71.2.72	strictVerticalSizing()	846
1.71.2.73	style()	846
1.71.2.74	styleClass()	846
1.71.2.75	unregisterBusy()	846
1.71.2.76	value()	847
1.71.2.77	verticalStretchAffinity()	847
1.71.2.78	visibility()	847
1.71.2.79	vstretch()	847
1.72	clove::ProxyDatasource Class Reference	848
1.72.1	Detailed Description	849
1.72.2	Member Function Documentation	849
1.72.2.1	changeValue()	849
1.72.2.2	columnCount()	849
1.72.2.3	columnHeadersVisible()	850

1.72.2.4	getColumnHeader()	850
1.72.2.5	getMetadata()	850
1.72.2.6	getRowHeader()	851
1.72.2.7	getValue()	851
1.72.2.8	nativePointerToProxyPointer()	851
1.72.2.9	OnDataInsert()	851
1.72.2.10	OnDataRemove()	852
1.72.2.11	OnDataUpdate()	852
1.72.2.12	OnHeaderDataInsert()	852
1.72.2.13	OnHeaderDataRemove()	852
1.72.2.14	OnHeaderDataUpdate()	852
1.72.2.15	OnHeaderVisibilityUpdated()	852
1.72.2.16	parent()	852
1.72.2.17	proxyPointerToNativePointer()	853
1.72.2.18	refresh()	853
1.72.2.19	rowCount()	853
1.72.2.20	rowHeadersVisible()	853
1.72.2.21	setDatasource()	854
1.72.2.22	valuePointer()	854
1.72.2.23	valuePointerNavigateInDepth()	854
1.73	clove::RadioButton Class Reference	855
1.73.1	Detailed Description	858
1.73.2	Member Function Documentation	858
1.73.2.1	addStyleClass()	858
1.73.2.2	bindProperty()	859
1.73.2.3	busy()	859
1.73.2.4	checked()	859
1.73.2.5	childrenWidgets()	859
1.73.2.6	computeMinimalHeightForWidth()	859
1.73.2.7	computeMinimalWidth()	860

1.73.2.8	<code>computePreferredHeightForWidth()</code>	860
1.73.2.9	<code>computePreferredWidth()</code>	860
1.73.2.10	<code>containingNameScope()</code>	860
1.73.2.11	<code>declareProperty()</code>	861
1.73.2.12	<code>doinit()</code>	861
1.73.2.13	<code>doinitEarly()</code>	861
1.73.2.14	<code>doresize()</code>	861
1.73.2.15	<code>doStandaloneResizing()</code>	862
1.73.2.16	<code>effectivelyEnabled()</code>	862
1.73.2.17	<code>effectiveVisibility()</code>	862
1.73.2.18	<code>enabled()</code>	862
1.73.2.19	<code>focus()</code>	862
1.73.2.20	<code>getMinimalHeightForWidth()</code>	862
1.73.2.21	<code>getMinimalWidth()</code>	863
1.73.2.22	<code>getPreferredHeightForWidth()</code>	863
1.73.2.23	<code>getPreferredWidth()</code>	863
1.73.2.24	<code>getProperty()</code>	863
1.73.2.25	<code>group()</code>	864
1.73.2.26	<code>groupValue()</code>	864
1.73.2.27	<code>hasFocus()</code>	864
1.73.2.28	<code>horizontalStretchAffinity()</code>	864
1.73.2.29	<code>hstretch()</code>	864
1.73.2.30	<code>init()</code>	864
1.73.2.31	<code>innerSize()</code>	865
1.73.2.32	<code>isAlive()</code>	865
1.73.2.33	<code>isStyleClass()</code>	865
1.73.2.34	<code>label()</code>	865
1.73.2.35	<code>mayFocus()</code>	865
1.73.2.36	<code>name()</code>	866
1.73.2.37	<code>nameScope()</code>	866

1.73.2.38 OnChanged()	866
1.73.2.39 OnClicked()	866
1.73.2.40 OnDestroyed()	866
1.73.2.41 OnKeyDown()	866
1.73.2.42 OnKeyPress()	867
1.73.2.43 OnKeyUp()	867
1.73.2.44 OnPropertyChanged()	867
1.73.2.45 OnResized()	867
1.73.2.46 OnVisibilityChanged()	867
1.73.2.47 outerSize()	868
1.73.2.48 parentWidget()	868
1.73.2.49 registerBusy()	868
1.73.2.50 relayout()	868
1.73.2.51 remove()	868
1.73.2.52 removeStyleClass()	869
1.73.2.53 resize()	869
1.73.2.54 setBusy()	869
1.73.2.55 setChecked()	869
1.73.2.56 setDoStandaloneResizing()	870
1.73.2.57 setEnabled()	870
1.73.2.58 setGroup()	870
1.73.2.59 setGroupValue()	871
1.73.2.60 setHorizontalStretchAffinity()	871
1.73.2.61 setHstretch()	871
1.73.2.62 setLabel()	871
1.73.2.63 setMayFocus()	872
1.73.2.64 setName()	872
1.73.2.65 setProperty()	872
1.73.2.66 setStrictHorizontalSizing()	873
1.73.2.67 setStrictVerticalSizing()	873

1.73.2.68	setStyle()	873
1.73.2.69	setStyleClass()	873
1.73.2.70	setStyleClassAssigned()	874
1.73.2.71	setValueDef()	874
1.73.2.72	setVerticalStretchAffinity()	874
1.73.2.73	setVisibility()	875
1.73.2.74	setVstretch()	875
1.73.2.75	strictHorizontalSizing()	875
1.73.2.76	strictVerticalSizing()	875
1.73.2.77	style()	875
1.73.2.78	styleClass()	876
1.73.2.79	unregisterBusy()	876
1.73.2.80	valueDef()	876
1.73.2.81	verticalStretchAffinity()	876
1.73.2.82	visibility()	876
1.73.2.83	vstretch()	877
1.74	clove::RadioGroup Class Reference	877
1.74.1	Detailed Description	877
1.74.2	Constructor & Destructor Documentation	877
1.74.2.1	RadioGroup()	878
1.74.3	Member Function Documentation	878
1.74.3.1	getGroupByPublicName()	878
1.74.3.2	OnChanged()	878
1.74.3.3	select()	878
1.74.3.4	selectByIndex()	878
1.74.3.5	selectByValue()	879
1.74.3.6	selected()	879
1.74.3.7	selectedIndex()	879
1.74.3.8	selectedValue()	879
1.74.3.9	unselectAll()	880

1.75	clove::RawResizeSplitter Class Reference	880
1.75.1	Detailed Description	883
1.75.2	Member Function Documentation	883
1.75.2.1	addStyleClass()	883
1.75.2.2	bindProperty()	883
1.75.2.3	busy()	884
1.75.2.4	childrenWidgets()	884
1.75.2.5	computeMinimalHeightForWidth()	884
1.75.2.6	computeMinimalWidth()	884
1.75.2.7	computePreferredHeightForWidth()	885
1.75.2.8	computePreferredWidth()	885
1.75.2.9	containingNameScope()	885
1.75.2.10	declareProperty()	885
1.75.2.11	doinit()	886
1.75.2.12	doinitEarly()	886
1.75.2.13	doresize()	886
1.75.2.14	doStandaloneResizing()	886
1.75.2.15	effectivelyEnabled()	886
1.75.2.16	effectiveVisibility()	887
1.75.2.17	enabled()	887
1.75.2.18	focus()	887
1.75.2.19	getMinimalHeightForWidth()	887
1.75.2.20	getMinimalWidth()	887
1.75.2.21	getPreferredHeightForWidth()	888
1.75.2.22	getPreferredWidth()	888
1.75.2.23	getProperty()	888
1.75.2.24	hasFocus()	888
1.75.2.25	horizontalStretchAffinity()	889
1.75.2.26	hstretch()	889
1.75.2.27	init()	889

1.75.2.28 innerSize()	889
1.75.2.29 isAlive()	889
1.75.2.30 isStyleClass()	890
1.75.2.31 mayFocus()	890
1.75.2.32 name()	890
1.75.2.33 nameScope()	890
1.75.2.34 OnDestroyed()	890
1.75.2.35 OnKeyDown()	891
1.75.2.36 OnKeyPress()	891
1.75.2.37 OnKeyUp()	891
1.75.2.38 OnMove()	891
1.75.2.39 OnMoveBegin()	891
1.75.2.40 OnMoveEnd()	891
1.75.2.41 OnPropertyChanged()	892
1.75.2.42 OnResized()	892
1.75.2.43 OnVisibilityChanged()	892
1.75.2.44 outerSize()	892
1.75.2.45 parentWidget()	892
1.75.2.46 registerBusy()	892
1.75.2.47 relayout()	893
1.75.2.48 remove()	893
1.75.2.49 removeStyleClass()	893
1.75.2.50 resize()	893
1.75.2.51 setBusy()	894
1.75.2.52 setDoStandaloneResizing()	894
1.75.2.53 setEnabled()	894
1.75.2.54 setHorizontalStretchAffinity()	894
1.75.2.55 setHstretch()	895
1.75.2.56 setMayFocus()	895
1.75.2.57 setName()	895

1.75.2.58	setProperty()	896
1.75.2.59	setStrictHorizontalSizing()	896
1.75.2.60	setStrictVerticalSizing()	896
1.75.2.61	setStyle()	896
1.75.2.62	setStyleClass()	897
1.75.2.63	setStyleClassAssigned()	897
1.75.2.64	setVerticalStretchAffinity()	897
1.75.2.65	setVisibility()	898
1.75.2.66	setVstretch()	898
1.75.2.67	strictHorizontalSizing()	898
1.75.2.68	strictVerticalSizing()	898
1.75.2.69	style()	898
1.75.2.70	styleClass()	899
1.75.2.71	unregisterBusy()	899
1.75.2.72	verticalStretchAffinity()	899
1.75.2.73	visibility()	899
1.75.2.74	vstretch()	899
1.76	clove::ResizeSplitter Class Reference	900
1.76.1	Detailed Description	903
1.76.2	Member Function Documentation	903
1.76.2.1	addBodyWidget()	903
1.76.2.2	addStyleClass()	904
1.76.2.3	bindProperty()	904
1.76.2.4	body()	904
1.76.2.5	bodyWidgets()	904
1.76.2.6	busy()	905
1.76.2.7	childrenWidgets()	905
1.76.2.8	computeMinimalHeightForWidth()	905
1.76.2.9	computeMinimalWidth()	905
1.76.2.10	computePreferredHeightForWidth()	905

1.76.2.11 computePreferredWidth()	906
1.76.2.12 containingNameScope()	906
1.76.2.13 declareProperty()	906
1.76.2.14 doinit()	906
1.76.2.15 doinitEarly()	907
1.76.2.16 doresize()	907
1.76.2.17 doStandaloneResizing()	907
1.76.2.18 effectivelyEnabled()	907
1.76.2.19 effectiveVisibility()	907
1.76.2.20 enabled()	908
1.76.2.21 focus()	908
1.76.2.22 getMinimalHeightForWidth()	908
1.76.2.23 getMinimalWidth()	908
1.76.2.24 getPreferredHeightForWidth()	908
1.76.2.25 getPreferredWidth()	909
1.76.2.26 getProperty()	909
1.76.2.27 hasFocus()	909
1.76.2.28 horizontalStretchAffinity()	909
1.76.2.29 hstretch()	910
1.76.2.30 init()	910
1.76.2.31 innerSize()	910
1.76.2.32 insertBodyWidget()	910
1.76.2.33 isAlive()	911
1.76.2.34 isStyleClass()	911
1.76.2.35 mayFocus()	911
1.76.2.36 name()	911
1.76.2.37 nameScope()	911
1.76.2.38 OnDestroyed()	912
1.76.2.39 OnKeyDown()	912
1.76.2.40 OnKeyPress()	912

1.76.2.41 OnKeyUp()	912
1.76.2.42 OnPropertyChanged()	912
1.76.2.43 OnResized()	912
1.76.2.44 OnVisibilityChanged()	913
1.76.2.45 orientation()	913
1.76.2.46 outerSize()	913
1.76.2.47 parentWidget()	913
1.76.2.48 registerBusy()	913
1.76.2.49 relayout()	913
1.76.2.50 remove()	913
1.76.2.51 removeStyleClass()	914
1.76.2.52 resize()	914
1.76.2.53 setBody()	914
1.76.2.54 setBodyWidget()	914
1.76.2.55 setBusy()	916
1.76.2.56 setDoStandaloneResizing()	916
1.76.2.57 setEnabled()	916
1.76.2.58 setHorizontalStretchAffinity()	917
1.76.2.59 setHstretch()	917
1.76.2.60 setMayFocus()	917
1.76.2.61 setName()	917
1.76.2.62 setOrientation()	918
1.76.2.63 setProperty()	918
1.76.2.64 setShowOnly()	918
1.76.2.65 setSizeFractions()	919
1.76.2.66 setSplitterVisibility()	919
1.76.2.67 setSplitterWidth()	919
1.76.2.68 setStrictHorizontalSizing()	919
1.76.2.69 setStrictVerticalSizing()	920
1.76.2.70 setStyle()	920

1.76.2.71	setStyleClass()	920
1.76.2.72	setStyleClassAssigned()	921
1.76.2.73	setVerticalStretchAffinity()	921
1.76.2.74	setVisibility()	921
1.76.2.75	setVstretch()	921
1.76.2.76	showOnly()	922
1.76.2.77	sizeFractions()	922
1.76.2.78	splitterVisibility()	922
1.76.2.79	splitterWidth()	922
1.76.2.80	strictHorizontalSizing()	922
1.76.2.81	strictVerticalSizing()	923
1.76.2.82	style()	923
1.76.2.83	styleClass()	923
1.76.2.84	unregisterBusy()	923
1.76.2.85	verticalStretchAffinity()	923
1.76.2.86	visibility()	924
1.76.2.87	vstretch()	924
1.77	clove::RichEdit Class Reference	924
1.77.1	Detailed Description	927
1.77.2	Member Function Documentation	927
1.77.2.1	addStyleClass()	927
1.77.2.2	bindProperty()	928
1.77.2.3	busy()	928
1.77.2.4	childrenWidgets()	928
1.77.2.5	computeMinimalHeightForWidth()	928
1.77.2.6	computeMinimalWidth()	929
1.77.2.7	computePreferredHeightForWidth()	929
1.77.2.8	computePreferredWidth()	929
1.77.2.9	containingNameScope()	929
1.77.2.10	declareProperty()	930

1.77.2.11 doinit()	930
1.77.2.12 doinitEarly()	930
1.77.2.13 doresize()	930
1.77.2.14 doStandaloneResizing()	931
1.77.2.15 effectivelyEnabled()	931
1.77.2.16 effectiveVisibility()	931
1.77.2.17 enabled()	931
1.77.2.18 focus()	931
1.77.2.19 getMinimalHeightForWidth()	931
1.77.2.20 getMinimalWidth()	932
1.77.2.21 getPreferredHeightForWidth()	932
1.77.2.22 getPreferredWidth()	932
1.77.2.23 getProperty()	932
1.77.2.24 hasFocus()	933
1.77.2.25 horizontalStretchAffinity()	933
1.77.2.26 hstretch()	933
1.77.2.27 htmlContent()	933
1.77.2.28 init()	933
1.77.2.29 innerSize()	934
1.77.2.30 isAlive()	934
1.77.2.31 isStyleClass()	934
1.77.2.32 mayFocus()	934
1.77.2.33 name()	934
1.77.2.34 nameScope()	935
1.77.2.35 OnDestroyed()	935
1.77.2.36 OnKeyDown()	935
1.77.2.37 OnKeyPress()	935
1.77.2.38 OnKeyUp()	935
1.77.2.39 OnPropertyChanged()	935
1.77.2.40 OnResized()	936

1.77.2.41 OnVisibilityChanged()	936
1.77.2.42 outerSize()	936
1.77.2.43 parentWidget()	936
1.77.2.44 registerBusy()	936
1.77.2.45 relayout()	936
1.77.2.46 remove()	936
1.77.2.47 removeStyleClass()	937
1.77.2.48 resize()	937
1.77.2.49 richeditbox()	937
1.77.2.50 setBusy()	937
1.77.2.51 setDoStandaloneResizing()	938
1.77.2.52 setEnabled()	938
1.77.2.53 setHorizontalStretchAffinity()	938
1.77.2.54 setHstretch()	938
1.77.2.55 setHtmlContent()	939
1.77.2.56 setMayFocus()	939
1.77.2.57 setName()	939
1.77.2.58 setProperty()	940
1.77.2.59 setStrictHorizontalSizing()	940
1.77.2.60 setStrictVerticalSizing()	940
1.77.2.61 setStyle()	940
1.77.2.62 setStyleClass()	941
1.77.2.63 setStyleClassAssigned()	941
1.77.2.64 setVerticalStretchAffinity()	941
1.77.2.65 setVisibility()	942
1.77.2.66 setVstretch()	942
1.77.2.67 strictHorizontalSizing()	942
1.77.2.68 strictVerticalSizing()	942
1.77.2.69 style()	942
1.77.2.70 styleClass()	943

1.77.2.71 unregisterBusy()	943
1.77.2.72 verticalStretchAffinity()	943
1.77.2.73 visibility()	943
1.77.2.74 vstretch()	943
1.78 clove::RichEditBox Class Reference	944
1.78.1 Detailed Description	947
1.78.2 Member Function Documentation	947
1.78.2.1 addStyleClass()	947
1.78.2.2 bindProperty()	948
1.78.2.3 busy()	948
1.78.2.4 childrenWidgets()	948
1.78.2.5 computeMinimalHeightForWidth()	948
1.78.2.6 computeMinimalWidth()	949
1.78.2.7 computePreferredHeightForWidth()	949
1.78.2.8 computePreferredWidth()	949
1.78.2.9 containingNameScope()	949
1.78.2.10 contentHtmlNode()	950
1.78.2.11 declareProperty()	950
1.78.2.12 doinit()	950
1.78.2.13 doinitEarly()	950
1.78.2.14 doresize()	951
1.78.2.15 doStandaloneResizing()	951
1.78.2.16 effectivelyEnabled()	951
1.78.2.17 effectiveVisibility()	951
1.78.2.18 enabled()	951
1.78.2.19 focus()	951
1.78.2.20 getMinimalHeightForWidth()	951
1.78.2.21 getMinimalWidth()	952
1.78.2.22 getPreferredHeightForWidth()	952
1.78.2.23 getPreferredWidth()	952

1.78.2.24	getProperty()	952
1.78.2.25	hasFocus()	953
1.78.2.26	horizontalStretchAffinity()	953
1.78.2.27	hstretch()	953
1.78.2.28	htmlContent()	953
1.78.2.29	init()	953
1.78.2.30	innerSize()	954
1.78.2.31	isAlive()	954
1.78.2.32	isStyleClass()	954
1.78.2.33	mayFocus()	954
1.78.2.34	name()	954
1.78.2.35	nameScope()	955
1.78.2.36	OnDestroyed()	955
1.78.2.37	OnKeyDown()	955
1.78.2.38	OnKeyPress()	955
1.78.2.39	OnKeyUp()	955
1.78.2.40	OnPropertyChanged()	955
1.78.2.41	OnResized()	956
1.78.2.42	OnVisibilityChanged()	956
1.78.2.43	outerSize()	956
1.78.2.44	parentWidget()	956
1.78.2.45	registerBusy()	956
1.78.2.46	relayout()	956
1.78.2.47	remove()	956
1.78.2.48	removeStyleClass()	957
1.78.2.49	resize()	957
1.78.2.50	selection()	957
1.78.2.51	selectionDecreaseFontSize()	957
1.78.2.52	selectionIncreaseFontSize()	958
1.78.2.53	selectionInsertList()	958

1.78.2.54 selectionSetBackgroundColor()	958
1.78.2.55 selectionSetForegroundColor()	958
1.78.2.56 setBusy()	959
1.78.2.57 setDoStandaloneResizing()	959
1.78.2.58 setEnabled()	959
1.78.2.59 setHorizontalStretchAffinity()	959
1.78.2.60 setHstretch()	960
1.78.2.61 setHtmlContent()	960
1.78.2.62 setMayFocus()	960
1.78.2.63 setName()	961
1.78.2.64 setProperty()	961
1.78.2.65 setSelection()	961
1.78.2.66 setStrictHorizontalSizing()	961
1.78.2.67 setStrictVerticalSizing()	962
1.78.2.68 setStyle()	962
1.78.2.69 setStyleClass()	962
1.78.2.70 setStyleClassAssigned()	963
1.78.2.71 setVerticalStretchAffinity()	963
1.78.2.72 setVisibility()	963
1.78.2.73 setVstretch()	963
1.78.2.74 strictHorizontalSizing()	964
1.78.2.75 strictVerticalSizing()	964
1.78.2.76 style()	964
1.78.2.77 styleClass()	964
1.78.2.78 toggleSelectionBold()	964
1.78.2.79 toggleSelectionItalic()	965
1.78.2.80 toggleSelectionUnderline()	965
1.78.2.81 unregisterBusy()	965
1.78.2.82 verticalStretchAffinity()	965
1.78.2.83 visibility()	965

1.78.2.84	vstretch()	966
1.79	clove::RootNameScope Class Reference	966
1.79.1	Detailed Description	966
1.79.2	Member Function Documentation	966
1.79.2.1	addChildNameScope()	966
1.79.2.2	getByName()	967
1.80	clove::ScrollView Class Reference	967
1.80.1	Detailed Description	970
1.80.2	Member Function Documentation	971
1.80.2.1	addStyleClass()	971
1.80.2.2	bindProperty()	971
1.80.2.3	body()	971
1.80.2.4	bodySize()	971
1.80.2.5	bodyWidget()	972
1.80.2.6	busy()	972
1.80.2.7	childrenWidgets()	972
1.80.2.8	computeMinimalHeightForWidth()	972
1.80.2.9	computeMinimalWidth()	972
1.80.2.10	computePreferredHeightForWidth()	973
1.80.2.11	computePreferredWidth()	973
1.80.2.12	containingNameScope()	973
1.80.2.13	declareProperty()	973
1.80.2.14	doinit()	974
1.80.2.15	doinitEarly()	974
1.80.2.16	doresize()	974
1.80.2.17	doStandaloneResizing()	974
1.80.2.18	effectivelyEnabled()	974
1.80.2.19	effectiveVisibility()	975
1.80.2.20	enabled()	975
1.80.2.21	focus()	975

1.80.2.22 getMinimalHeightForWidth()	975
1.80.2.23 getMinimalWidth()	975
1.80.2.24 getPreferredHeightForWidth()	976
1.80.2.25 getPreferredWidth()	976
1.80.2.26 getProperty()	976
1.80.2.27 hasFocus()	976
1.80.2.28 horizontalScrollPosition()	977
1.80.2.29 horizontalStretchAffinity()	977
1.80.2.30 hstretch()	977
1.80.2.31 init()	977
1.80.2.32 innerSize()	977
1.80.2.33 isAlive()	978
1.80.2.34 isStyleClass()	978
1.80.2.35 mayFocus()	978
1.80.2.36 name()	978
1.80.2.37 nameScope()	978
1.80.2.38 OnDestroyed()	979
1.80.2.39 OnKeyDown()	979
1.80.2.40 OnKeyPress()	979
1.80.2.41 OnKeyUp()	979
1.80.2.42 OnPropertyChanged()	979
1.80.2.43 OnResized()	979
1.80.2.44 OnVisibilityChanged()	980
1.80.2.45 outerSize()	980
1.80.2.46 parentWidget()	980
1.80.2.47 registerBusy()	980
1.80.2.48 relayout()	980
1.80.2.49 remove()	980
1.80.2.50 removeStyleClass()	981
1.80.2.51 resize()	981

1.80.2.52	setBody()	981
1.80.2.53	setBusy()	981
1.80.2.54	setDoStandaloneResizing()	982
1.80.2.55	setEnabled()	982
1.80.2.56	setHorizontalScrollPosition()	982
1.80.2.57	setHorizontalStretchAffinity()	983
1.80.2.58	setHstretch()	983
1.80.2.59	setMayFocus()	983
1.80.2.60	setName()	983
1.80.2.61	setProperty()	984
1.80.2.62	setStrictHorizontalSizing()	984
1.80.2.63	setStrictVerticalSizing()	984
1.80.2.64	setStyle()	985
1.80.2.65	setStyleClass()	985
1.80.2.66	setStyleClassAssigned()	985
1.80.2.67	setVerticalScrollPosition()	985
1.80.2.68	setVerticalStretchAffinity()	987
1.80.2.69	setVisibility()	987
1.80.2.70	setVstretch()	987
1.80.2.71	strictHorizontalSizing()	988
1.80.2.72	strictVerticalSizing()	988
1.80.2.73	style()	988
1.80.2.74	styleClass()	988
1.80.2.75	unregisterBusy()	988
1.80.2.76	verticalScrollPosition()	989
1.80.2.77	verticalStretchAffinity()	989
1.80.2.78	visibility()	989
1.80.2.79	vstretch()	989
1.81	clove::Slider Class Reference	989
1.81.1	Detailed Description	993

1.81.2	Member Function Documentation	993
1.81.2.1	addStyleClass()	993
1.81.2.2	bindProperty()	993
1.81.2.3	busy()	994
1.81.2.4	childrenWidgets()	994
1.81.2.5	computeMinimalHeightForWidth()	994
1.81.2.6	computeMinimalWidth()	994
1.81.2.7	computePreferredHeightForWidth()	995
1.81.2.8	computePreferredWidth()	995
1.81.2.9	containingNameScope()	995
1.81.2.10	declareProperty()	995
1.81.2.11	doinit()	996
1.81.2.12	doinitEarly()	996
1.81.2.13	doresize()	996
1.81.2.14	doStandaloneResizing()	996
1.81.2.15	effectivelyEnabled()	996
1.81.2.16	effectiveVisibility()	997
1.81.2.17	enabled()	997
1.81.2.18	focus()	997
1.81.2.19	getMinimalHeightForWidth()	997
1.81.2.20	getMinimalWidth()	997
1.81.2.21	getPreferredHeightForWidth()	998
1.81.2.22	getPreferredWidth()	998
1.81.2.23	getProperty()	998
1.81.2.24	hasFocus()	998
1.81.2.25	horizontalStretchAffinity()	999
1.81.2.26	hstretch()	999
1.81.2.27	init()	999
1.81.2.28	innerSize()	999
1.81.2.29	isAlive()	999

1.81.2.30 isStyleClass()	1000
1.81.2.31 max()	1000
1.81.2.32 mayFocus()	1000
1.81.2.33 min()	1000
1.81.2.34 name()	1000
1.81.2.35 nameScope()	1001
1.81.2.36 OnChanged()	1001
1.81.2.37 OnDestroyed()	1001
1.81.2.38 OnKeyDown()	1001
1.81.2.39 OnKeyPress()	1001
1.81.2.40 OnKeyUp()	1001
1.81.2.41 OnPropertyChanged()	1002
1.81.2.42 OnResized()	1002
1.81.2.43 OnVisibilityChanged()	1002
1.81.2.44 orientation()	1002
1.81.2.45 outerSize()	1002
1.81.2.46 parentWidget()	1002
1.81.2.47 registerBusy()	1003
1.81.2.48 relayout()	1003
1.81.2.49 remove()	1003
1.81.2.50 removeStyleClass()	1003
1.81.2.51 resize()	1004
1.81.2.52 setBusy()	1004
1.81.2.53 setDoStandaloneResizing()	1004
1.81.2.54 setEnabled()	1004
1.81.2.55 setHorizontalStretchAffinity()	1005
1.81.2.56 setHstretch()	1005
1.81.2.57 setMax()	1005
1.81.2.58 setMayFocus()	1005
1.81.2.59 setMin()	1006

1.81.2.60 setName()	1006
1.81.2.61 setOrientation()	1006
1.81.2.62 setProperty()	1007
1.81.2.63 setStepsize()	1007
1.81.2.64 setStrictHorizontalSizing()	1007
1.81.2.65 setStrictVerticalSizing()	1007
1.81.2.66 setStyle()	1008
1.81.2.67 setStyleClass()	1008
1.81.2.68 setStyleClassAssigned()	1008
1.81.2.69 setValue()	1009
1.81.2.70 setVerticalStretchAffinity()	1009
1.81.2.71 setVisibility()	1009
1.81.2.72 setVstretch()	1009
1.81.2.73 stepsize()	1010
1.81.2.74 strictHorizontalSizing()	1010
1.81.2.75 strictVerticalSizing()	1010
1.81.2.76 style()	1010
1.81.2.77 styleClass()	1010
1.81.2.78 unregisterBusy()	1010
1.81.2.79 value()	1011
1.81.2.80 verticalStretchAffinity()	1011
1.81.2.81 visibility()	1011
1.81.2.82 vstretch()	1011
1.82 clove::SortProxyDatasource Class Reference	1012
1.82.1 Detailed Description	1013
1.82.2 Constructor & Destructor Documentation	1013
1.82.2.1 SortProxyDatasource()	1013
1.82.3 Member Function Documentation	1013
1.82.3.1 changeValue()	1014
1.82.3.2 columnCount()	1014

1.82.3.3	columnHeadersVisible()	1014
1.82.3.4	getColumnHeader()	1014
1.82.3.5	getMetadata()	1016
1.82.3.6	getRowHeader()	1016
1.82.3.7	getValue()	1016
1.82.3.8	nativePointerToProxyPointer()	1017
1.82.3.9	OnDataInsert()	1017
1.82.3.10	OnDataRemove()	1017
1.82.3.11	OnDataUpdate()	1017
1.82.3.12	OnHeaderDataInsert()	1017
1.82.3.13	OnHeaderDataRemove()	1018
1.82.3.14	OnHeaderDataUpdate()	1018
1.82.3.15	OnHeaderVisibilityUpdated()	1018
1.82.3.16	parent()	1018
1.82.3.17	proxyPointerToNativePointer()	1018
1.82.3.18	refresh()	1019
1.82.3.19	rowCount()	1019
1.82.3.20	rowHeadersVisible()	1019
1.82.3.21	setColumnComparator()	1019
1.82.3.22	setDatasource()	1020
1.82.3.23	setRowComparator()	1020
1.82.3.24	valuePointer()	1020
1.82.3.25	valuePointerNavigateInDepth()	1020
1.83	clove::Spacer Class Reference	1021
1.83.1	Detailed Description	1024
1.83.2	Member Function Documentation	1024
1.83.2.1	addStyleClass()	1024
1.83.2.2	bindProperty()	1025
1.83.2.3	busy()	1025
1.83.2.4	childrenWidgets()	1025

1.83.2.5	<code>computeMinimalHeightForWidth()</code>	1025
1.83.2.6	<code>computeMinimalWidth()</code>	1026
1.83.2.7	<code>computePreferredHeightForWidth()</code>	1026
1.83.2.8	<code>computePreferredWidth()</code>	1026
1.83.2.9	<code>containingNameScope()</code>	1026
1.83.2.10	<code>declareProperty()</code>	1026
1.83.2.11	<code>doinit()</code>	1027
1.83.2.12	<code>doinitEarly()</code>	1027
1.83.2.13	<code>doresize()</code>	1027
1.83.2.14	<code>doStandaloneResizing()</code>	1027
1.83.2.15	<code>effectivelyEnabled()</code>	1028
1.83.2.16	<code>effectiveVisibility()</code>	1028
1.83.2.17	<code>enabled()</code>	1028
1.83.2.18	<code>focus()</code>	1028
1.83.2.19	<code>getMinimalHeightForWidth()</code>	1028
1.83.2.20	<code>getMinimalWidth()</code>	1029
1.83.2.21	<code>getPreferredHeightForWidth()</code>	1029
1.83.2.22	<code>getPreferredWidth()</code>	1029
1.83.2.23	<code>getProperty()</code>	1029
1.83.2.24	<code>hasFocus()</code>	1030
1.83.2.25	<code>horizontalStretchAffinity()</code>	1030
1.83.2.26	<code>hstretch()</code>	1030
1.83.2.27	<code>init()</code>	1030
1.83.2.28	<code>innerSize()</code>	1030
1.83.2.29	<code>isAlive()</code>	1031
1.83.2.30	<code>isStyleClass()</code>	1031
1.83.2.31	<code>mayFocus()</code>	1031
1.83.2.32	<code>name()</code>	1031
1.83.2.33	<code>nameScope()</code>	1031
1.83.2.34	<code>OnDestroyed()</code>	1032

1.83.2.35 OnKeyDown()	1032
1.83.2.36 OnKeyPress()	1032
1.83.2.37 OnKeyUp()	1032
1.83.2.38 OnPropertyChanged()	1032
1.83.2.39 OnResized()	1032
1.83.2.40 OnVisibilityChanged()	1033
1.83.2.41 outerSize()	1033
1.83.2.42 parentWidget()	1033
1.83.2.43 registerBusy()	1033
1.83.2.44 relayout()	1033
1.83.2.45 remove()	1033
1.83.2.46 removeStyleClass()	1034
1.83.2.47 resize()	1034
1.83.2.48 setBusy()	1034
1.83.2.49 setDoStandaloneResizing()	1034
1.83.2.50 setEnabled()	1035
1.83.2.51 setHorizontalStretchAffinity()	1035
1.83.2.52 setHstretch()	1035
1.83.2.53 setMayFocus()	1036
1.83.2.54 setName()	1036
1.83.2.55 setProperty()	1036
1.83.2.56 setStrictHorizontalSizing()	1036
1.83.2.57 setStrictVerticalSizing()	1037
1.83.2.58 setStyle()	1037
1.83.2.59 setStyleClass()	1037
1.83.2.60 setStyleClassAssigned()	1038
1.83.2.61 setVerticalStretchAffinity()	1038
1.83.2.62 setVisibility()	1038
1.83.2.63 setVstretch()	1038
1.83.2.64 strictHorizontalSizing()	1039

1.83.2.65 strictVerticalSizing()	1039
1.83.2.66 style()	1039
1.83.2.67 styleClass()	1039
1.83.2.68 unregisterBusy()	1039
1.83.2.69 verticalStretchAffinity()	1040
1.83.2.70 visibility()	1040
1.83.2.71 vstretch()	1040
1.84 clove::StackLayout Class Reference	1040
1.84.1 Detailed Description	1041
1.84.2 Member Function Documentation	1041
1.84.2.1 addChild()	1041
1.84.2.2 children()	1041
1.84.2.3 clearChilds()	1042
1.84.2.4 insertColumn()	1042
1.84.2.5 insertRow()	1042
1.84.2.6 removeColumn()	1042
1.84.2.7 removeRow()	1043
1.84.2.8 setChildren()	1043
1.85 clove::symbols Class Reference	1043
1.85.1 Detailed Description	1043
1.86 clove::TableView Class Reference	1044
1.86.1 Detailed Description	1048
1.86.2 Member Function Documentation	1049
1.86.2.1 addStyleClass()	1049
1.86.2.2 allowChecking()	1049
1.86.2.3 allowSelection()	1049
1.86.2.4 alwaysAllocateExpanderSpace()	1049
1.86.2.5 bindProperty()	1049
1.86.2.6 busy()	1050
1.86.2.7 cellGenerator()	1050

1.86.2.8	checkedCells()	1050
1.86.2.9	childrenWidgets()	1050
1.86.2.10	collapseCell()	1050
1.86.2.11	columnsResizable()	1051
1.86.2.12	computeMinimalHeightForWidth()	1051
1.86.2.13	computeMinimalWidth()	1051
1.86.2.14	computePreferredHeightForWidth()	1051
1.86.2.15	computePreferredWidth()	1052
1.86.2.16	containingNameScope()	1052
1.86.2.17	datasource()	1052
1.86.2.18	dataViewProcessor()	1052
1.86.2.19	declareProperty()	1052
1.86.2.20	doinit()	1053
1.86.2.21	doinitEarly()	1053
1.86.2.22	doresize()	1053
1.86.2.23	doStandaloneResizing()	1053
1.86.2.24	editCell()	1053
1.86.2.25	editOnGesture()	1054
1.86.2.26	effectivelyEnabled()	1054
1.86.2.27	effectiveVisibility()	1054
1.86.2.28	enabled()	1054
1.86.2.29	expandCell()	1054
1.86.2.30	expandCellRecursive()	1055
1.86.2.31	focus()	1055
1.86.2.32	getMinimalHeightForWidth()	1055
1.86.2.33	getMinimalWidth()	1055
1.86.2.34	getPreferredHeightForWidth()	1056
1.86.2.35	getPreferredWidth()	1056
1.86.2.36	getProperty()	1056
1.86.2.37	granularity()	1056

1.86.2.38 gridVisible()	1057
1.86.2.39 hasFocus()	1057
1.86.2.40 headersource()	1057
1.86.2.41 hideExpanders()	1057
1.86.2.42 horizontalStretchAffinity()	1057
1.86.2.43 hstretch()	1057
1.86.2.44 init()	1057
1.86.2.45 innerSize()	1058
1.86.2.46 isAlive()	1058
1.86.2.47 isCellChecked()	1058
1.86.2.48 isCellExpanded()	1058
1.86.2.49 isCellSelected()	1059
1.86.2.50 isStyleClass()	1059
1.86.2.51 mayFocus()	1059
1.86.2.52 name()	1059
1.86.2.53 nameScope()	1059
1.86.2.54 nodeActivationNeedsDoubleClick()	1060
1.86.2.55 OnDestroyed()	1060
1.86.2.56 OnKeyDown()	1060
1.86.2.57 OnKeyPress()	1060
1.86.2.58 OnKeyUp()	1060
1.86.2.59 OnPropertyChanged()	1060
1.86.2.60 OnResized()	1061
1.86.2.61 OnSelectionChanged()	1061
1.86.2.62 OnVisibilityChanged()	1061
1.86.2.63 outerSize()	1061
1.86.2.64 parentWidget()	1061
1.86.2.65 registerBusy()	1061
1.86.2.66 relayout()	1062
1.86.2.67 remove()	1062

1.86.2.68 removeStyleClass()	1062
1.86.2.69 resize()	1062
1.86.2.70 rowsResizable()	1063
1.86.2.71 selectCell()	1063
1.86.2.72 selection()	1063
1.86.2.73 setAllowChecking()	1063
1.86.2.74 setAllowSelection()	1063
1.86.2.75 setAlwaysAllocateExpanderSpace()	1064
1.86.2.76 setBusy()	1064
1.86.2.77 setCellChecked()	1064
1.86.2.78 setCellGenerator()	1065
1.86.2.79 setCheckedCells()	1065
1.86.2.80 setColumnsResizable()	1065
1.86.2.81 setDatasource()	1065
1.86.2.82 setDataViewProcessor()	1066
1.86.2.83 setDoStandaloneResizing()	1066
1.86.2.84 setEditOnGesture()	1066
1.86.2.85 setEnabled()	1067
1.86.2.86 setGranularity()	1067
1.86.2.87 setGridVisible()	1067
1.86.2.88 setHeadersource()	1067
1.86.2.89 setHideExpanders()	1068
1.86.2.90 setHorizontalStretchAffinity()	1068
1.86.2.91 setHstretch()	1068
1.86.2.92 setMayFocus()	1069
1.86.2.93 setName()	1069
1.86.2.94 setNodeActivationNeedsDoubleClick()	1069
1.86.2.95 setProperty()	1069
1.86.2.96 setRowsResizable()	1070
1.86.2.97 setShowChangeMenu()	1070

1.86.2.98	setShowOnlyFirstColumn()	1070
1.86.2.99	setStrictHorizontalSizing()	1071
1.86.2.100	setStrictVerticalSizing()	1071
1.86.2.101	setStyle()	1071
1.86.2.102	setStyleClass()	1071
1.86.2.103	setStyleClassAssigned()	1072
1.86.2.104	setVerticalStretchAffinity()	1072
1.86.2.105	setVisibility()	1072
1.86.2.106	setVstretch()	1073
1.86.2.107	showChangeMenu()	1073
1.86.2.108	showOnlyFirstColumn()	1073
1.86.2.109	strictHorizontalSizing()	1073
1.86.2.110	strictVerticalSizing()	1074
1.86.2.111	style()	1074
1.86.2.112	styleClass()	1074
1.86.2.113	unregisterBusy()	1074
1.86.2.114	verticalStretchAffinity()	1074
1.86.2.115	visibility()	1075
1.86.2.116	vstretch()	1075
1.87	clove::TabView Class Reference	1075
1.87.1	Detailed Description	1079
1.87.2	Member Function Documentation	1079
1.87.2.1	addStyleClass()	1079
1.87.2.2	addTab()	1079
1.87.2.3	bindProperty()	1080
1.87.2.4	busy()	1080
1.87.2.5	childrenWidgets()	1080
1.87.2.6	computeMinimalHeightForWidth()	1080
1.87.2.7	computeMinimalWidth()	1081
1.87.2.8	computePreferredHeightForWidth()	1081

1.87.2.9	<code>computePreferredWidth()</code>	1081
1.87.2.10	<code>containingNameScope()</code>	1081
1.87.2.11	<code>currentTab()</code>	1082
1.87.2.12	<code>declareProperty()</code>	1082
1.87.2.13	<code>doinit()</code>	1082
1.87.2.14	<code>doinitEarly()</code>	1082
1.87.2.15	<code>doresize()</code>	1083
1.87.2.16	<code>doStandaloneResizing()</code>	1083
1.87.2.17	<code>effectivelyEnabled()</code>	1083
1.87.2.18	<code>effectiveVisibility()</code>	1083
1.87.2.19	<code>enabled()</code>	1083
1.87.2.20	<code>focus()</code>	1083
1.87.2.21	<code>getMinimalHeightForWidth()</code>	1083
1.87.2.22	<code>getMinimalWidth()</code>	1084
1.87.2.23	<code>getPreferredHeightForWidth()</code>	1084
1.87.2.24	<code>getPreferredWidth()</code>	1084
1.87.2.25	<code>getProperty()</code>	1084
1.87.2.26	<code>hasFocus()</code>	1085
1.87.2.27	<code>horizontalStretchAffinity()</code>	1085
1.87.2.28	<code>hstretch()</code>	1085
1.87.2.29	<code>init()</code>	1085
1.87.2.30	<code>innerSize()</code>	1086
1.87.2.31	<code>isAlive()</code>	1086
1.87.2.32	<code>isStyleClass()</code>	1086
1.87.2.33	<code>mayFocus()</code>	1086
1.87.2.34	<code>name()</code>	1086
1.87.2.35	<code>nameScope()</code>	1087
1.87.2.36	<code>OnDestroyed()</code>	1087
1.87.2.37	<code>OnKeyDown()</code>	1087
1.87.2.38	<code>OnKeyPress()</code>	1087

1.87.2.39 OnKeyUp()	1087
1.87.2.40 OnPropertyChanged()	1087
1.87.2.41 OnResized()	1088
1.87.2.42 OnTabCreationRequested()	1088
1.87.2.43 OnVisibilityChanged()	1088
1.87.2.44 outerSize()	1088
1.87.2.45 parentWidget()	1088
1.87.2.46 registerBusy()	1088
1.87.2.47 relayout()	1089
1.87.2.48 remove()	1089
1.87.2.49 removeStyleClass()	1089
1.87.2.50 resize()	1089
1.87.2.51 setBusy()	1090
1.87.2.52 setCurrentTab()	1090
1.87.2.53 setDoStandaloneResizing()	1090
1.87.2.54 setEnabled()	1090
1.87.2.55 setHorizontalStretchAffinity()	1091
1.87.2.56 setHstretch()	1091
1.87.2.57 setMayFocus()	1091
1.87.2.58 setName()	1092
1.87.2.59 setProperty()	1092
1.87.2.60 setStrictHorizontalSizing()	1092
1.87.2.61 setStrictVerticalSizing()	1092
1.87.2.62 setStyle()	1093
1.87.2.63 setStyleClass()	1093
1.87.2.64 setStyleClassAssigned()	1093
1.87.2.65 setSwitchInvisibleAnimationDuration()	1094
1.87.2.66 setSwitchInvisibleAnimationName()	1094
1.87.2.67 setSwitchVisibleAnimationDuration()	1094
1.87.2.68 setSwitchVisibleAnimationName()	1094

1.87.2.69	setTabBarLocation()	1095
1.87.2.70	setTabs()	1095
1.87.2.71	setUserMayAddTabs()	1095
1.87.2.72	setVerticalStretchAffinity()	1096
1.87.2.73	setVisibility()	1096
1.87.2.74	setVstretch()	1096
1.87.2.75	strictHorizontalSizing()	1096
1.87.2.76	strictVerticalSizing()	1097
1.87.2.77	style()	1097
1.87.2.78	styleClass()	1097
1.87.2.79	switchInvisibleAnimationDuration()	1097
1.87.2.80	switchInvisibleAnimationName()	1097
1.87.2.81	switchVisibleAnimationDuration()	1097
1.87.2.82	switchVisibleAnimationName()	1098
1.87.2.83	tabBarLocation()	1098
1.87.2.84	tabs()	1098
1.87.2.85	unregisterBusy()	1098
1.87.2.86	userMayAddTabs()	1098
1.87.2.87	verticalStretchAffinity()	1099
1.87.2.88	visibility()	1099
1.87.2.89	vstretch()	1099
1.88	clove::TimeBox Class Reference	1099
1.88.1	Detailed Description	1103
1.88.2	Member Function Documentation	1103
1.88.2.1	addStyleClass()	1103
1.88.2.2	autocompletionFilter()	1104
1.88.2.3	autocompletionItems()	1104
1.88.2.4	autocompletionOpenForNoText()	1104
1.88.2.5	bindProperty()	1104
1.88.2.6	busy()	1104

1.88.2.7	childrenWidgets()	1105
1.88.2.8	computeMinimalHeightForWidth()	1105
1.88.2.9	computeMinimalWidth()	1105
1.88.2.10	computePreferredHeightForWidth()	1105
1.88.2.11	computePreferredWidth()	1105
1.88.2.12	containingNameScope()	1106
1.88.2.13	declareProperty()	1106
1.88.2.14	doinit()	1106
1.88.2.15	doinitEarly()	1106
1.88.2.16	doresize()	1107
1.88.2.17	doStandaloneResizing()	1107
1.88.2.18	effectivelyEnabled()	1107
1.88.2.19	effectiveVisibility()	1107
1.88.2.20	enabled()	1107
1.88.2.21	focus()	1107
1.88.2.22	getMinimalHeightForWidth()	1107
1.88.2.23	getMinimalWidth()	1108
1.88.2.24	getPreferredHeightForWidth()	1108
1.88.2.25	getPreferredWidth()	1108
1.88.2.26	getProperty()	1108
1.88.2.27	hasFocus()	1109
1.88.2.28	hintText()	1109
1.88.2.29	horizontalStretchAffinity()	1109
1.88.2.30	hstretch()	1109
1.88.2.31	init()	1109
1.88.2.32	innerSize()	1110
1.88.2.33	isAlive()	1110
1.88.2.34	isStyleClass()	1110
1.88.2.35	mayFocus()	1110
1.88.2.36	name()	1110

1.88.2.37	nameScope()	1111
1.88.2.38	OnChanged()	1111
1.88.2.39	OnDestroyed()	1111
1.88.2.40	OnKeyDown()	1111
1.88.2.41	OnKeyPress()	1111
1.88.2.42	OnKeyUp()	1111
1.88.2.43	OnPopupTextSelected()	1112
1.88.2.44	OnPropertyChanged()	1112
1.88.2.45	OnResized()	1112
1.88.2.46	OnVisibilityChanged()	1112
1.88.2.47	outerSize()	1112
1.88.2.48	parentWidget()	1112
1.88.2.49	readOnly()	1113
1.88.2.50	registerBusy()	1113
1.88.2.51	relayout()	1113
1.88.2.52	remove()	1113
1.88.2.53	removeStyleClass()	1113
1.88.2.54	resize()	1114
1.88.2.55	setAutocompletionFilter()	1114
1.88.2.56	setAutocompletionItems()	1114
1.88.2.57	setAutocompletionOpenForNoText()	1114
1.88.2.58	setBusy()	1115
1.88.2.59	setDoStandaloneResizing()	1115
1.88.2.60	setEnabled()	1115
1.88.2.61	setHintText()	1116
1.88.2.62	setHorizontalStretchAffinity()	1116
1.88.2.63	setHstretch()	1116
1.88.2.64	setMayFocus()	1116
1.88.2.65	setName()	1117
1.88.2.66	setProperty()	1117

1.88.2.67	setReadOnly()	1117
1.88.2.68	setStrictHorizontalSizing()	1118
1.88.2.69	setStrictVerticalSizing()	1118
1.88.2.70	setStyle()	1118
1.88.2.71	setStyleClass()	1118
1.88.2.72	setStyleClassAssigned()	1119
1.88.2.73	setText()	1119
1.88.2.74	setTextSelection()	1119
1.88.2.75	setTime()	1120
1.88.2.76	setVerticalStretchAffinity()	1120
1.88.2.77	setVisibility()	1120
1.88.2.78	setVstretch()	1120
1.88.2.79	strictHorizontalSizing()	1121
1.88.2.80	strictVerticalSizing()	1121
1.88.2.81	style()	1121
1.88.2.82	styleClass()	1121
1.88.2.83	text()	1121
1.88.2.84	textSelection()	1122
1.88.2.85	time()	1122
1.88.2.86	unregisterBusy()	1122
1.88.2.87	verticalStretchAffinity()	1122
1.88.2.88	visibility()	1122
1.88.2.89	vstretch()	1123
1.89	clove::Toolbar Class Reference	1123
1.89.1	Detailed Description	1126
1.89.2	Member Function Documentation	1126
1.89.2.1	actions()	1127
1.89.2.2	addStyleClass()	1127
1.89.2.3	bindProperty()	1127
1.89.2.4	busy()	1127

1.89.2.5	childrenWidgets()	1127
1.89.2.6	computeMinimalHeightForWidth()	1128
1.89.2.7	computeMinimalWidth()	1128
1.89.2.8	computePreferredHeightForWidth()	1128
1.89.2.9	computePreferredWidth()	1128
1.89.2.10	containingNameScope()	1129
1.89.2.11	declareProperty()	1129
1.89.2.12	doinit()	1129
1.89.2.13	doinitEarly()	1129
1.89.2.14	doresize()	1130
1.89.2.15	doStandaloneResizing()	1130
1.89.2.16	effectivelyEnabled()	1130
1.89.2.17	effectiveVisibility()	1130
1.89.2.18	enabled()	1130
1.89.2.19	focus()	1130
1.89.2.20	getMinimalHeightForWidth()	1130
1.89.2.21	getMinimalWidth()	1131
1.89.2.22	getPreferredHeightForWidth()	1131
1.89.2.23	getPreferredWidth()	1131
1.89.2.24	getProperty()	1131
1.89.2.25	hasFocus()	1132
1.89.2.26	head1()	1132
1.89.2.27	head2()	1132
1.89.2.28	headControl()	1132
1.89.2.29	headControlWidget()	1132
1.89.2.30	horizontalStretchAffinity()	1133
1.89.2.31	hstretch()	1133
1.89.2.32	icon()	1133
1.89.2.33	init()	1133
1.89.2.34	innerSize()	1133

1.89.2.35 <code>isAlive()</code>	1134
1.89.2.36 <code>isStyleClass()</code>	1134
1.89.2.37 <code>mayFocus()</code>	1134
1.89.2.38 <code>name()</code>	1134
1.89.2.39 <code>nameScope()</code>	1134
1.89.2.40 <code>OnActionTriggered()</code>	1135
1.89.2.41 <code>OnDestroyed()</code>	1135
1.89.2.42 <code>OnKeyDown()</code>	1135
1.89.2.43 <code>OnKeyPress()</code>	1135
1.89.2.44 <code>OnKeyUp()</code>	1135
1.89.2.45 <code>OnPropertyChanged()</code>	1135
1.89.2.46 <code>OnResized()</code>	1136
1.89.2.47 <code>OnVisibilityChanged()</code>	1136
1.89.2.48 <code>outerSize()</code>	1136
1.89.2.49 <code>parentWidget()</code>	1136
1.89.2.50 <code>registerBusy()</code>	1136
1.89.2.51 <code>relayout()</code>	1136
1.89.2.52 <code>remove()</code>	1136
1.89.2.53 <code>removeStyleClass()</code>	1137
1.89.2.54 <code>resize()</code>	1137
1.89.2.55 <code>setActions()</code>	1137
1.89.2.56 <code>setBusy()</code>	1137
1.89.2.57 <code>setDoStandaloneResizing()</code>	1138
1.89.2.58 <code>setEnabled()</code>	1138
1.89.2.59 <code>setHead1()</code>	1138
1.89.2.60 <code>setHead2()</code>	1139
1.89.2.61 <code>setHeadControl()</code>	1139
1.89.2.62 <code>setHorizontalStretchAffinity()</code>	1139
1.89.2.63 <code>setHstretch()</code>	1139
1.89.2.64 <code>setIcon()</code>	1140

1.89.2.65 setMayFocus()	1140
1.89.2.66 setName()	1140
1.89.2.67 setProperty()	1141
1.89.2.68 setStrictHorizontalSizing()	1141
1.89.2.69 setStrictVerticalSizing()	1141
1.89.2.70 setStyle()	1141
1.89.2.71 setStyleClass()	1142
1.89.2.72 setStyleClassAssigned()	1142
1.89.2.73 setVerticalStretchAffinity()	1142
1.89.2.74 setVisibility()	1143
1.89.2.75 setVstretch()	1143
1.89.2.76 strictHorizontalSizing()	1143
1.89.2.77 strictVerticalSizing()	1143
1.89.2.78 style()	1143
1.89.2.79 styleClass()	1144
1.89.2.80 unregisterBusy()	1144
1.89.2.81 verticalStretchAffinity()	1144
1.89.2.82 visibility()	1144
1.89.2.83 vstretch()	1144
1.90 clove::TreeView Class Reference	1145
1.90.1 Detailed Description	1149
1.90.2 Member Function Documentation	1150
1.90.2.1 addStyleClass()	1150
1.90.2.2 allowChecking()	1150
1.90.2.3 allowSelection()	1150
1.90.2.4 alwaysAllocateExpanderSpace()	1150
1.90.2.5 bindProperty()	1150
1.90.2.6 busy()	1151
1.90.2.7 cellGenerator()	1151
1.90.2.8 checkedCells()	1151

1.90.2.9	childrenWidgets()	1151
1.90.2.10	collapseCell()	1151
1.90.2.11	columnsResizable()	1152
1.90.2.12	computeMinimalHeightForWidth()	1152
1.90.2.13	computeMinimalWidth()	1152
1.90.2.14	computePreferredHeightForWidth()	1152
1.90.2.15	computePreferredWidth()	1153
1.90.2.16	containingNameScope()	1153
1.90.2.17	datasource()	1153
1.90.2.18	dataViewProcessor()	1153
1.90.2.19	declareProperty()	1153
1.90.2.20	doinit()	1154
1.90.2.21	doinitEarly()	1154
1.90.2.22	doresize()	1154
1.90.2.23	doStandaloneResizing()	1154
1.90.2.24	editCell()	1154
1.90.2.25	editOnGesture()	1155
1.90.2.26	effectivelyEnabled()	1155
1.90.2.27	effectiveVisibility()	1155
1.90.2.28	enabled()	1155
1.90.2.29	expandCell()	1155
1.90.2.30	expandCellRecursive()	1156
1.90.2.31	focus()	1156
1.90.2.32	getMinimalHeightForWidth()	1156
1.90.2.33	getMinimalWidth()	1156
1.90.2.34	getPreferredHeightForWidth()	1157
1.90.2.35	getPreferredWidth()	1157
1.90.2.36	getProperty()	1157
1.90.2.37	granularity()	1157
1.90.2.38	gridVisible()	1158

1.90.2.39	hasFocus()	1158
1.90.2.40	headersource()	1158
1.90.2.41	hideExpanders()	1158
1.90.2.42	horizontalStretchAffinity()	1158
1.90.2.43	hstretch()	1158
1.90.2.44	init()	1158
1.90.2.45	innerSize()	1159
1.90.2.46	isAlive()	1159
1.90.2.47	isCellChecked()	1159
1.90.2.48	isCellExpanded()	1159
1.90.2.49	isCellSelected()	1160
1.90.2.50	isStyleClass()	1160
1.90.2.51	mayFocus()	1160
1.90.2.52	name()	1160
1.90.2.53	nameScope()	1160
1.90.2.54	nodeActivationNeedsDoubleClick()	1161
1.90.2.55	OnDestroyed()	1161
1.90.2.56	OnKeyDown()	1161
1.90.2.57	OnKeyPress()	1161
1.90.2.58	OnKeyUp()	1161
1.90.2.59	OnPropertyChanged()	1161
1.90.2.60	OnResized()	1162
1.90.2.61	OnSelectionChanged()	1162
1.90.2.62	OnVisibilityChanged()	1162
1.90.2.63	outerSize()	1162
1.90.2.64	parentWidget()	1162
1.90.2.65	registerBusy()	1162
1.90.2.66	relayout()	1163
1.90.2.67	remove()	1163
1.90.2.68	removeStyleClass()	1163

1.90.2.69	resize()	1163
1.90.2.70	rowsResizable()	1164
1.90.2.71	selectCell()	1164
1.90.2.72	selection()	1164
1.90.2.73	setAllowChecking()	1164
1.90.2.74	setAllowSelection()	1164
1.90.2.75	setAlwaysAllocateExpanderSpace()	1165
1.90.2.76	setBusy()	1165
1.90.2.77	setCellChecked()	1165
1.90.2.78	setCellGenerator()	1166
1.90.2.79	setCheckedCells()	1166
1.90.2.80	setColumnsResizable()	1166
1.90.2.81	setDatasource()	1166
1.90.2.82	setDataViewProcessor()	1167
1.90.2.83	setDoStandaloneResizing()	1167
1.90.2.84	setEditOnGesture()	1167
1.90.2.85	setEnabled()	1168
1.90.2.86	setGranularity()	1168
1.90.2.87	setGridVisible()	1168
1.90.2.88	setHeadersource()	1168
1.90.2.89	setHideExpanders()	1169
1.90.2.90	setHorizontalStretchAffinity()	1169
1.90.2.91	setHstretch()	1169
1.90.2.92	setMayFocus()	1170
1.90.2.93	setName()	1170
1.90.2.94	setNodeActivationNeedsDoubleClick()	1170
1.90.2.95	setProperty()	1170
1.90.2.96	setRowsResizable()	1171
1.90.2.97	setShowChangeMenu()	1171
1.90.2.98	setShowOnlyFirstColumn()	1171

1.90.2.99	setStrictHorizontalSizing()	1172
1.90.2.100	setStrictVerticalSizing()	1172
1.90.2.101	setStyle()	1172
1.90.2.102	setStyleClass()	1172
1.90.2.103	setStyleClassAssigned()	1173
1.90.2.104	setVerticalStretchAffinity()	1173
1.90.2.105	setVisibility()	1173
1.90.2.106	setVstretch()	1174
1.90.2.107	showChangeMenu()	1174
1.90.2.108	showOnlyFirstColumn()	1174
1.90.2.109	strictHorizontalSizing()	1174
1.90.2.110	strictVerticalSizing()	1175
1.90.2.111	style()	1175
1.90.2.112	styleClass()	1175
1.90.2.113	unregisterBusy()	1175
1.90.2.114	verticalStretchAffinity()	1175
1.90.2.115	visibility()	1176
1.90.2.116	vstretch()	1176
1.91	clove::utils Class Reference	1176
1.91.1	Detailed Description	1177
1.91.2	Member Function Documentation	1177
1.91.2.1	addOnDestroyHandler()	1177
1.91.2.2	addOnDoubleClickTapHandler()	1178
1.91.2.3	addOnDragHandler()	1178
1.91.2.4	animateNode()	1179
1.91.2.5	applyDefaultsToConfig()	1179
1.91.2.6	arraysum()	1179
1.91.2.7	arrayToDatasource()	1180
1.91.2.8	connectDatasourceToWidget()	1180
1.91.2.9	cssLengthToPixels()	1181

1.91.2.10	deferLoading()	1181
1.91.2.11	findWidgetForDomNode()	1182
1.91.2.12	getExtraSize()	1182
1.91.2.13	getFullPathForLoadedResource()	1182
1.91.2.14	getInnerSize()	1183
1.91.2.15	getOuterSize()	1183
1.91.2.16	insertToArray()	1183
1.91.2.17	isDescendantOf()	1184
1.91.2.18	main_parts_resized()	1184
1.91.2.19	OnMainResized()	1184
1.91.2.20	popup()	1184
1.91.2.21	removeOnDoubleClickTapHandler()	1185
1.91.2.22	resumeResizing()	1185
1.91.2.23	runOnLoaded()	1185
1.91.2.24	setStyleClassAssigned()	1186
1.91.2.25	suspendResizing()	1186
1.91.2.26	tryLoadScript()	1186
1.91.2.27	tryLoadStyle()	1187
1.92	clove::VerticalStack Class Reference	1187
1.92.1	Detailed Description	1191
1.92.2	Member Function Documentation	1191
1.92.2.1	addChild()	1191
1.92.2.2	addStyleClass()	1191
1.92.2.3	bindProperty()	1192
1.92.2.4	busy()	1192
1.92.2.5	children()	1192
1.92.2.6	childrenWidgets()	1192
1.92.2.7	clearChilds()	1192
1.92.2.8	computeMinimalHeightForWidth()	1192
1.92.2.9	computeMinimalWidth()	1193

1.92.2.10 computePreferredHeightForWidth()	1193
1.92.2.11 computePreferredWidth()	1193
1.92.2.12 containingNameScope()	1193
1.92.2.13 declareProperty()	1194
1.92.2.14 doinit()	1194
1.92.2.15 doinitEarly()	1194
1.92.2.16 doresize()	1194
1.92.2.17 doStandaloneResizing()	1195
1.92.2.18 effectivelyEnabled()	1195
1.92.2.19 effectiveVisibility()	1195
1.92.2.20 enabled()	1195
1.92.2.21 focus()	1195
1.92.2.22 getMinimalHeightForWidth()	1195
1.92.2.23 getMinimalWidth()	1196
1.92.2.24 getPreferredHeightForWidth()	1196
1.92.2.25 getPreferredWidth()	1196
1.92.2.26 getProperty()	1196
1.92.2.27 hasFocus()	1197
1.92.2.28 horizontalStretchAffinity()	1197
1.92.2.29 hstretch()	1197
1.92.2.30 init()	1197
1.92.2.31 innerSize()	1198
1.92.2.32 insertColumn()	1198
1.92.2.33 insertRow()	1198
1.92.2.34 isAlive()	1198
1.92.2.35 isStyleClass()	1198
1.92.2.36 mayFocus()	1199
1.92.2.37 name()	1199
1.92.2.38 nameScope()	1199
1.92.2.39 OnDestroyed()	1199

1.92.2.40 OnKeyDown()	1199
1.92.2.41 OnKeyPress()	1200
1.92.2.42 OnKeyUp()	1200
1.92.2.43 OnPropertyChanged()	1200
1.92.2.44 OnResized()	1200
1.92.2.45 OnVisibilityChanged()	1200
1.92.2.46 outerSize()	1201
1.92.2.47 parentWidget()	1201
1.92.2.48 registerBusy()	1201
1.92.2.49 relayout()	1201
1.92.2.50 remove()	1201
1.92.2.51 removeColumn()	1202
1.92.2.52 removeRow()	1202
1.92.2.53 removeStyleClass()	1202
1.92.2.54 resize()	1203
1.92.2.55 rows()	1203
1.92.2.56 setBusy()	1203
1.92.2.57 setChildren()	1203
1.92.2.58 setDoStandaloneResizing()	1203
1.92.2.59 setEnabled()	1205
1.92.2.60 setHorizontalStretchAffinity()	1205
1.92.2.61 setHstretch()	1205
1.92.2.62 setMayFocus()	1206
1.92.2.63 setName()	1206
1.92.2.64 setProperty()	1206
1.92.2.65 setRows()	1206
1.92.2.66 setStrictHorizontalSizing()	1207
1.92.2.67 setStrictVerticalSizing()	1207
1.92.2.68 setStyle()	1207
1.92.2.69 setStyleClass()	1208

1.92.2.70	setStyleClassAssigned()	1208
1.92.2.71	setVerticalStretchAffinity()	1208
1.92.2.72	setVisibility()	1208
1.92.2.73	setVstretch()	1209
1.92.2.74	strictHorizontalSizing()	1209
1.92.2.75	strictVerticalSizing()	1209
1.92.2.76	style()	1209
1.92.2.77	styleClass()	1210
1.92.2.78	unregisterBusy()	1210
1.92.2.79	verticalStretchAffinity()	1210
1.92.2.80	visibility()	1210
1.92.2.81	vstretch()	1210
1.93	clove::VideoPlayer Class Reference	1211
1.93.1	Detailed Description	1215
1.93.2	Member Function Documentation	1215
1.93.2.1	addStyleClass()	1215
1.93.2.2	autoPlay()	1215
1.93.2.3	bindProperty()	1216
1.93.2.4	busy()	1216
1.93.2.5	canPlayType()	1216
1.93.2.6	childrenWidgets()	1216
1.93.2.7	computeMinimalHeightForWidth()	1217
1.93.2.8	computeMinimalWidth()	1218
1.93.2.9	computePreferredHeightForWidth()	1218
1.93.2.10	computePreferredWidth()	1218
1.93.2.11	containingNameScope()	1218
1.93.2.12	currentMediaPosition()	1219
1.93.2.13	declareProperty()	1219
1.93.2.14	doinit()	1219
1.93.2.15	doinitEarly()	1219

1.93.2.16 doresize()	1220
1.93.2.17 doStandaloneResizing()	1220
1.93.2.18 duration()	1220
1.93.2.19 effectivelyEnabled()	1220
1.93.2.20 effectiveVisibility()	1220
1.93.2.21 enabled()	1220
1.93.2.22 focus()	1221
1.93.2.23 getMinimalHeightForWidth()	1221
1.93.2.24 getMinimalWidth()	1222
1.93.2.25 getPreferredHeightForWidth()	1222
1.93.2.26 getPreferredWidth()	1222
1.93.2.27 getProperty()	1222
1.93.2.28 hasEnded()	1223
1.93.2.29 hasFocus()	1223
1.93.2.30 horizontalStretchAffinity()	1223
1.93.2.31 hstretch()	1223
1.93.2.32 init()	1223
1.93.2.33 innerSize()	1224
1.93.2.34 isAlive()	1224
1.93.2.35 isPaused()	1224
1.93.2.36 isStyleClass()	1224
1.93.2.37 loop()	1224
1.93.2.38 mayFocus()	1225
1.93.2.39 muted()	1225
1.93.2.40 name()	1225
1.93.2.41 nameScope()	1225
1.93.2.42 nativeNode()	1225
1.93.2.43 OnDestroyed()	1225
1.93.2.44 OnDurationChanged()	1226
1.93.2.45 OnHasEnded()	1226

1.93.2.46 OnIsPaused()	1226
1.93.2.47 OnIsPlaying()	1226
1.93.2.48 OnKeyDown()	1226
1.93.2.49 OnKeyPress()	1227
1.93.2.50 OnKeyUp()	1227
1.93.2.51 OnLoadingAborted()	1227
1.93.2.52 OnPropertyChanged()	1227
1.93.2.53 OnReadyForPlayback()	1227
1.93.2.54 OnResized()	1227
1.93.2.55 OnVisibilityChanged()	1228
1.93.2.56 outerSize()	1228
1.93.2.57 parentWidget()	1228
1.93.2.58 pause()	1228
1.93.2.59 play()	1228
1.93.2.60 preload()	1228
1.93.2.61 registerBusy()	1229
1.93.2.62 relayout()	1229
1.93.2.63 remove()	1229
1.93.2.64 removeStyleClass()	1229
1.93.2.65 resize()	1230
1.93.2.66 setAutoPlay()	1230
1.93.2.67 setBusy()	1230
1.93.2.68 setCurrentMediaPosition()	1230
1.93.2.69 setDoStandaloneResizing()	1231
1.93.2.70 setEnabled()	1231
1.93.2.71 setHorizontalStretchAffinity()	1231
1.93.2.72 setHstretch()	1231
1.93.2.73 setLoop()	1232
1.93.2.74 setMayFocus()	1232
1.93.2.75 setMuted()	1232

1.93.2.76 setName()	1233
1.93.2.77 setPreload()	1233
1.93.2.78 setProperty()	1233
1.93.2.79 setShowControls()	1233
1.93.2.80 setSource()	1234
1.93.2.81 setStrictHorizontalSizing()	1234
1.93.2.82 setStrictVerticalSizing()	1234
1.93.2.83 setStyle()	1235
1.93.2.84 setStyleClass()	1235
1.93.2.85 setStyleClassAssigned()	1235
1.93.2.86 setVerticalStretchAffinity()	1235
1.93.2.87 setVisibility()	1236
1.93.2.88 setVolume()	1236
1.93.2.89 setVstretch()	1236
1.93.2.90 showControls()	1237
1.93.2.91 source()	1237
1.93.2.92 strictHorizontalSizing()	1237
1.93.2.93 strictVerticalSizing()	1237
1.93.2.94 style()	1237
1.93.2.95 styleClass()	1237
1.93.2.96 unregisterBusy()	1237
1.93.2.97 verticalStretchAffinity()	1238
1.93.2.98 visibility()	1238
1.93.2.99 volume()	1238
1.93.2.100vstretch()	1238
1.94 clove::Widget Class Reference	1239
1.94.1 Detailed Description	1242
1.94.2 Constructor & Destructor Documentation	1242
1.94.2.1 Widget()	1242
1.94.3 Member Function Documentation	1243

1.94.3.1	addStyleClass()	1243
1.94.3.2	bindProperty()	1243
1.94.3.3	busy()	1243
1.94.3.4	childrenWidgets()	1244
1.94.3.5	computeMinimalHeightForWidth()	1244
1.94.3.6	computeMinimalWidth()	1244
1.94.3.7	computePreferredHeightForWidth()	1244
1.94.3.8	computePreferredWidth()	1244
1.94.3.9	containingNameScope()	1245
1.94.3.10	declareProperty()	1245
1.94.3.11	doinit()	1245
1.94.3.12	doinitEarly()	1245
1.94.3.13	doresize()	1246
1.94.3.14	doStandaloneResizing()	1246
1.94.3.15	effectivelyEnabled()	1246
1.94.3.16	effectiveVisibility()	1246
1.94.3.17	enabled()	1246
1.94.3.18	focus()	1246
1.94.3.19	getMinimalHeightForWidth()	1246
1.94.3.20	getMinimalWidth()	1247
1.94.3.21	getPreferredHeightForWidth()	1247
1.94.3.22	getPreferredWidth()	1247
1.94.3.23	getProperty()	1247
1.94.3.24	hasFocus()	1248
1.94.3.25	horizontalStretchAffinity()	1248
1.94.3.26	hstretch()	1248
1.94.3.27	init()	1248
1.94.3.28	innerSize()	1249
1.94.3.29	isAlive()	1249
1.94.3.30	isStyleClass()	1249

1.94.3.31	mayFocus()	1249
1.94.3.32	name()	1249
1.94.3.33	nameScope()	1250
1.94.3.34	OnDestroyed()	1250
1.94.3.35	OnKeyDown()	1250
1.94.3.36	OnKeyPress()	1250
1.94.3.37	OnKeyUp()	1250
1.94.3.38	OnPropertyChanged()	1250
1.94.3.39	OnResized()	1251
1.94.3.40	OnVisibilityChanged()	1251
1.94.3.41	outerSize()	1251
1.94.3.42	parentWidget()	1251
1.94.3.43	registerBusy()	1251
1.94.3.44	relayout()	1251
1.94.3.45	remove()	1251
1.94.3.46	removeStyleClass()	1252
1.94.3.47	resize()	1252
1.94.3.48	setBusy()	1252
1.94.3.49	setDoStandaloneResizing()	1252
1.94.3.50	setEnabled()	1253
1.94.3.51	setHorizontalStretchAffinity()	1253
1.94.3.52	setHstretch()	1253
1.94.3.53	setMayFocus()	1254
1.94.3.54	setName()	1254
1.94.3.55	setProperty()	1254
1.94.3.56	setStrictHorizontalSizing()	1254
1.94.3.57	setStrictVerticalSizing()	1255
1.94.3.58	setStyle()	1255
1.94.3.59	setStyleClass()	1255
1.94.3.60	setStyleClassAssigned()	1256

1.94.3.61	setVerticalStretchAffinity()	1256
1.94.3.62	setVisibility()	1256
1.94.3.63	setVstretch()	1256
1.94.3.64	strictHorizontalSizing()	1257
1.94.3.65	strictVerticalSizing()	1257
1.94.3.66	style()	1257
1.94.3.67	styleClass()	1257
1.94.3.68	unregisterBusy()	1257
1.94.3.69	verticalStretchAffinity()	1258
1.94.3.70	visibility()	1258
1.94.3.71	vstretch()	1258
1.95	clove::Wrap Class Reference	1258
1.95.1	Detailed Description	1262
1.95.2	Member Function Documentation	1262
1.95.2.1	addChild()	1262
1.95.2.2	addStyleClass()	1262
1.95.2.3	bindProperty()	1262
1.95.2.4	busy()	1263
1.95.2.5	children()	1263
1.95.2.6	childrenWidgets()	1263
1.95.2.7	clearChilds()	1263
1.95.2.8	computeMinimalHeightForWidth()	1263
1.95.2.9	computeMinimalWidth()	1264
1.95.2.10	computePreferredHeightForWidth()	1264
1.95.2.11	computePreferredWidth()	1264
1.95.2.12	containingNameScope()	1264
1.95.2.13	declareProperty()	1264
1.95.2.14	doinit()	1265
1.95.2.15	doinitEarly()	1265
1.95.2.16	doresize()	1265

1.95.2.17 doStandaloneResizing()	1265
1.95.2.18 effectivelyEnabled()	1266
1.95.2.19 effectiveVisibility()	1266
1.95.2.20 enabled()	1266
1.95.2.21 focus()	1266
1.95.2.22 getMinimalHeightForWidth()	1266
1.95.2.23 getMinimalWidth()	1267
1.95.2.24 getPreferredHeightForWidth()	1267
1.95.2.25 getPreferredWidth()	1267
1.95.2.26 getProperty()	1267
1.95.2.27 hasFocus()	1268
1.95.2.28 horizontalStretchAffinity()	1268
1.95.2.29 hstretch()	1268
1.95.2.30 init()	1268
1.95.2.31 innerSize()	1268
1.95.2.32 isAlive()	1269
1.95.2.33 isStyleClass()	1269
1.95.2.34 mayFocus()	1269
1.95.2.35 name()	1269
1.95.2.36 nameScope()	1269
1.95.2.37 OnDestroyed()	1270
1.95.2.38 OnKeyDown()	1270
1.95.2.39 OnKeyPress()	1270
1.95.2.40 OnKeyUp()	1270
1.95.2.41 OnPropertyChanged()	1270
1.95.2.42 OnResized()	1270
1.95.2.43 OnVisibilityChanged()	1271
1.95.2.44 outerSize()	1271
1.95.2.45 parentWidget()	1271
1.95.2.46 registerBusy()	1271

1.95.2.47	relayout()	1271
1.95.2.48	remove()	1271
1.95.2.49	removeStyleClass()	1272
1.95.2.50	resize()	1272
1.95.2.51	setBusy()	1272
1.95.2.52	setChildren()	1272
1.95.2.53	setDoStandaloneResizing()	1273
1.95.2.54	setEnabled()	1273
1.95.2.55	setHorizontalStretchAffinity()	1273
1.95.2.56	setHstretch()	1274
1.95.2.57	setMayFocus()	1274
1.95.2.58	setName()	1274
1.95.2.59	setProperty()	1274
1.95.2.60	setStrictHorizontalSizing()	1275
1.95.2.61	setStrictVerticalSizing()	1275
1.95.2.62	setStyle()	1275
1.95.2.63	setStyleClass()	1276
1.95.2.64	setStyleClassAssigned()	1276
1.95.2.65	setVerticalStretchAffinity()	1276
1.95.2.66	setVisibility()	1276
1.95.2.67	setVstretch()	1277
1.95.2.68	strictHorizontalSizing()	1277
1.95.2.69	strictVerticalSizing()	1277
1.95.2.70	style()	1277
1.95.2.71	styleClass()	1278
1.95.2.72	unregisterBusy()	1278
1.95.2.73	verticalStretchAffinity()	1278
1.95.2.74	visibility()	1278
1.95.2.75	vstretch()	1278
1.96	clove::WrapLayout Class Reference	1279
1.96.1	Detailed Description	1279
1.96.2	Member Function Documentation	1279
1.96.2.1	addChild()	1279
1.96.2.2	children()	1280
1.96.2.3	clearChilds()	1280
1.96.2.4	setChildren()	1280

Chapter 1

Clove Manual

1.1 License

clove is written by Josef Hahn under the terms of the GPLv3 or higher.

Please read the `LICENSE` file from the package and the [Dependencies](#) section for included third-party stuff.

1.2 About

Clove is a user interface library for the web, which offers a powerful browser-side JavaScript toolkit for composing rich and neat web application frontends.

Clove is designed to get work done.

- **Integrating Clove is easy!** Just one or two `scripts` and one `style` to be added to your existing html before you can begin.
- **The programming interface is easy!** No esoteric tricks for simple things. It's 100% standard JavaScript, running inside the browser, with a clean api.
- **A rich and complete set of widgets is included.** This goes from labels and buttons to data views like tables and trees. It's easy to implement custom widgets.
- All widgets can be used **either stand-alone in your existing document flow or completely managed** by the powerful Clove layouting system.
- **It works in all modern web browsers**, mobile and desktop, and can help realizing applications working great in both worlds.
- Beyond widgets, Clove comes with many kind of versatile tools for typical tasks, like internationalization, branding, ...
- It's well documented. Read the Manual for the first steps, to lookup some stuff later on, and for the api reference.
- There are live demos available, which can be opened in the web browser without any preparation needed.
- It's a true **Free Software** project without commercial pro-versions.

1.3 Up-to-date?

Are you currently reading from another source than the homepage? Are you in doubt if that place is up-to-date? If yes, you should visit <https://pseudopolis.eu/wiki/pino/projs/clove> and check that. You are currently reading the manual for version 0.2.1242.

1.4 Maturity

In this version, the state of clove is considered as production-stable.

1.5 Dependencies

There are external parts which are used by clove. Many thanks to the projects and all participants.

banner image *included*: `_meta/homepage_bannerimage.png`; public domain; copied from [here](#).

jquery *included*: licensed under terms of [GPLv2](#).

1.6 Introduction

Clove is a browser-side JavaScript library which mostly provides graphical widgets. In order to use Clove, you have to include it into your web page:

```
<!doctype html>
<html>
  <head>
    ...
    <script type="text/javascript" src="jquery.js"></script>
    <script type="text/javascript" src="clove.js"></script>
    ...
  </head>
  ...
</html>
```

The `jquery.js` can either be the included one or a different one, which is compatible. It contains the jQuery library, which is a 3rd-party component required by Clove.

There is also `clove.css` needed by Clove, but this is loaded automatically.

1.7 First Steps

After you have [included Clove in your web page](#), you can use its programming interface in some way in order to support your application.

The easiest and most straight-forward way is to start with an entirely empty web page (i.e. nothing in its `body`) and to include a script, which bootstraps your application frontend:

```
<!doctype html>
<html>
  <head>
    <script type="text/javascript" src="jquery.js"></script>
    <script type="text/javascript" src="clove.js"></script>
    <script type="text/javascript">
      clove.build({
        view: "MainView",
        head1: "My first Clove application",
        body: {view: "Label", label: "Hello World!"},
      });
    </script>
  </head>
  <body>
  </body>
</html>
```

The interesting part is the `clove.build` call, which constructs the user interface according to a 'blueprint'. The blueprint is a JavaScript object, which contains some key/value pairs, specifying details about the user interface. In this example we see a blueprint for a `MainView`, with some properties specified (including `body`, whose value is again a blueprint for an inner widget).

The following sections will explain more details about `clove.build` and the configuration values you can specify.

Please **Note**: The pattern of bundling many configuration values into one object - typically written as `{k1:v1, ...}` - and just using this object as a parameter is ubiquitous in Clove. It is not only used for widget configurations as here, but also at other places where many parameters are optional and/or vary depending on the context. **Whenever the documentation describes a parameter as a 'configuration object', or sometimes even just 'configuration', it is about such an object.** The documentation explains all relevant details about such objects for the affected functions. *This is something like 'optional parameters' for JavaScript.*

1.8 How To Build User Interfaces

In the next parts, we will see how to build a user interface in some more depth and how to use it for actual user interactions.

1.8.1 Building Widgets

As we have seen, user interfaces are built with the `clove.build` function. There are a few other functions, which also build user interfaces; in an equal way, but adapted for some specific situations. Directly constructing widgets by call the class constructor is not allowed. The `clove.build` call has the following general form:

```
clove.build({...}, {...});
```

The only two parameters it has are configuration objects (as mentioned above). The first one is a widget configuration (called 'blueprint' in the beginning). The second one is a build configuration, which controls some construction aspects. It is optional and not interesting in the beginning, while the first one is essential.

```
clove.build({view: 'Label'});
```

This is a very small widget configuration. Each widget configuration at least contains a `view` parameter, which specifies the widget class to instantiate. So this declaration controls what kind of widget to build.

Please **Note**: Although this is true enough for the moment, we will also see widget configurations without a `view` parameter later on. This is just because it can be implicitly clear in some situations.

Depending on the widget class, several other parameters exist. All of them are optional, but some are essential for making any real use of the widget. The `Label` widget shows just a piece of text; the `label` parameter of it specifies this text:

```
clove.build({view: "Label", label: "Hello World!"});
```

A few parameters are available for all widgets, as we will see [later on](#), while many others come from the specific classes. Whenever you are interested in the properties available for a particular class, or whenever you want to learn more about any other part of the programming interface, read the relevant parts in the Developer Documentation.

For an overview of the existing widget classes, please see the *WidgetShowroom* demo. You can find the demos in your package, e.g. `clove/_meta/demos/WidgetShowroom/index.html`, or on the Clove homepage.

Please **Note**: The entire Clove programming interface resides in `clove.`, so the complete way to refer e.g. to the `label` class is `clove.Label`. The `view` parameter allows to omit the `clove.` though.

Some widgets are containers, which host one or more inner widgets. Those inner widgets are specified in configuration parameters as well, so a natural nesting results:

```
clove.build({
  view: "MainView",
  head1: "My first Clove application",
  body: {
    view: "Label",
    label: "Hello World!",
  },
});
```

This builds a widget of class `MainView`. This class can host one inner widget specified in `body`. This nesting can go arbitrarily deep and is used to structure complete user interfaces in one call.

`clove.populateUI`

This section closes with the introduction of a tool, which is not required but recommended to use. The `clove.populateUI` function enwraps time-consuming actions, as building a complex user interface can be, and gives the user some better feedback.

You should enwrap a complex `clove.build` call with it this way:

```
clove.populateUI(function() {
  clove.build({
    view: ...
  });
});
```

1.8.2 Layout And Alignment

Building complex user interfaces is possible with container widgets. Technically they are just normal widgets, but they have properties (i.e. configuration keys in a widget configuration) for child widgets. Some of them are rather special-purpose, providing some real functionality as well. Others are just intended for grouping other child widgets together and align them in some defined way.

The latter group is usually called 'layouts'. There are a few layout types, all implementing a different but general-purpose behavior of alignment for its child widgets.

1.8.2.1 Stack Layout

A stack layout can be either horizontal or vertical. A horizontal layout aligns its child widgets column-based, side-by-side, all with the full height. A vertical one aligns row-based, all with the full width respectively. They are configured this way:

```
clove.build({
  view: "VerticalStack",
  rows: [
    {view: "Label", label: "Hello"},
    {view: "Label", label: "... my friend!"},
  ],
});
```

A horizontal layout is implemented by `HorizontalStack` and uses the `cols` property for its childs.

Please **Note**: There is a shortcut for horizontal and vertical stacks, which is commonly used: The `view` parameter may be omitted.

A larger example:

```
clove.build({
  rows: [
    {cols: [
      {view: "Label", label: "a1"},
      {view: "Label", label: "b1"},
      {view: "Label", label: "c1"},
    ]},
    {cols: [
      {view: "Label", label: "a2"},
      {view: "Label", label: "b2"},
      {view: "Label", label: "c2"},
    ]},
    {cols: [
      {view: "Label", label: "a3"},
      {view: "Label", label: "b3"},
      {view: "Label", label: "c3"},
    ]},
  ],
});
```

See also the *HeinersAsiaShop* demo.

1.8.2.2 Grid Layout

The last example tries to realize a 3x3 grid of labels. But as a grid it would not work great, because the column widths aren't connected. A real grid helps out here:

```
clove.build({
  view: "Grid",
  children: [
    {view: "Label", label: "a1", row: 0, col: 0},
    {view: "Label", label: "b1", row: 0, col: 1},
    {view: "Label", label: "c1", row: 0, col: 2},
    {view: "Label", label: "a2", row: 1, col: 0},
    {view: "Label", label: "b2", row: 1, col: 1},
    {view: "Label", label: "c2", row: 1, col: 2},
    {view: "Label", label: "a3", row: 2, col: 0},
    {view: "Label", label: "b3", row: 2, col: 1},
    {view: "Label", label: "c3", row: 2, col: 2},
  ],
});
```

1.8.2.3 Finetuning

There are some ways to finetune the alignment within such layouts. They make sense isolated or in some combinations in many situations.

Stretch Affinities

A layout has to divide the available space on one or both axes. In the first place this is done by asking the widgets for a preferred size. If more space is available, it can distribute it among the childs in a configurable way.

Assigning numbers to the childs' `horizontalStretchAffinity` and/or `verticalStretchAffinity` properties will lead to a distribution of free space in this exact proportions.

```
clove.build({
  cols: [
    {view: "Label", label: "Foo", horizontalStretchAffinity: 2},
    {view: "Label", label: "Bar", horizontalStretchAffinity: 1},
    {view: "Label", label: "Baz", /*horizontalStretchAffinity: 0*/},
  ],
});
```

Custom Sizes

The preferred size of child widgets is not the right choice in each situation. Sometimes it is required to set something fixed (and e.g. use scroll views if needed) in order to make the composition work. For such cases, specify `width` and/or `height` for some childs (as css length).

```
clove.build({
  cols: [
    {view: "Label", label: "Foo", width: "50pt"},
    {view: "SomethingDifferent", ..., height: "50pt"},
    {view: "Label", label: "Bar", width: "50pt"},
  ],
});
```

Stretching

Typically a widget uses all the space it can get from the layout. It bids for getting additional space by means of the `horizontalStretchAffinity`/`verticalStretchAffinity` properties. Even if it is set to 0, there is no guarantee to not get additional space.

The `strictHorizontalSizing` and `strictVerticalSizing` properties control if a widget is just placed within that space or if it actually fills the additional size.

```
clove.build({
  cols: [
    {view: "Label", label: "Foo", strictVerticalSizing: true},
    /* ... */
  ],
});
```

1.9 Widget Names

After constructing a user interface, you probably will need to access some of the widgets for working with its content or state. Since you specified just the blueprint, you don't have any direct access to the constructed widgets.

The easiest way is to assign a `name` to interesting widgets and to get the actual widget instances by those names afterwards:

```
clove.build({
  cols: [
    {view: "Label", label: "Foo", name: "foolabel"},
    {view: "Label", label: "Bar", name: "barlabel"},
  ],
});
```

After executing this build call, use the `clove.getByName` method for getting the widget instances:

```
var foolabel = clove.getByName("foolabel");
// do something with it ...
foolabel.setLabel("Fuh");
```

1.9.1 Name Scopes

The model behind widget naming is called 'name scopes'. They are indeed a bit more complex than just `getByName`. There can be many separated partitions of names. This avoids name conflicts between different contexts. On the other hand, a namespace can have child scopes, which are automatically included in name lookups.

When it comes to [Custom Widgets](#) we will learn more, but for the moment we can use namespaces whenever we build multiple Clove user interface and want to avoid name conflicts. This is an advanced topic you can also skip at first.

For a new namespace, just construct `clove.RootNameScope` and use it in the build configuration (this is the second parameter of `clove.build`):

```
var mynamespace = clove.RootNameScope();
clove.build({
  rows: ...
}, {
  nameScope: mynamespace,
});
```

Each name used in the widget configuration is now known in `mynamespace`. If you would execute such code at two different places, you would end up with two isolated namespaces and no conflicts.

Getting the widgets is now possible with the namespace object:

```
var foolabel = mynamespace.getByName("foolabel");
```

1.10 Common Widget Functionality

The common base class for all widgets is `clove.Widget`. It implements some common functionality which is automatically available for each widget. All of its subclasses can be build and configured as introduced above. A later chapter will also explain how to implement [Custom Widgets](#).

1.10.1 Widget Property System

The most essential thing all widgets have in common is the property system. It provides the foundation for all ways of working with the widgets' contents and states.

The root of this system is a key/value map of property values. The key is a simple description string like 'label', the value can be an arbitrary JavaScript object. This map is used for the complete configuration and state information of a widget instance.

You can access them by [clove.Widget.getProperty](#) and [clove.Widget.setProperty](#):

```
var foolabel = clove.getByName("foolabel");
foolabel.setProperty("label", "Fuh");
```

While you can set values for arbitrary keys, some of them will be understood by the widget implementation and processed in some way. The `label` property is understood by a [clove.Label](#), so the last example actually makes sense.

Those properties are the same as the one you set in a widget configuration, so it's the same `label` property as we already have seen in former examples (analogously you could overwrite the `rows` property of a `VerticalStack` with new child configurations) - but this way it is possible to dynamically change and retrieve those values.

For the well-known properties of a class, there is always a dedicated getter and setter as well:

```
var s = foolabel.label() + "xyz";
foolabel.setLabel(s);
```

1.10.2 Common Widget Properties

On top of this property system, the [clove.Widget](#) class provides some common properties (see the Developer Documentation for details):

- `visibility`: The visibility of a widget.
- `enabled`: If a widget is enabled, i.e. can interact with the user.
- `styleClass`: Additional css classes for styling. See [Styling](#) for more.

1.10.3 Common Widget Management

There are also some common management methods, i.e. for removing widgets. See the Developer Documentation for them.

1.11 Events

The Clove event system is mostly built by the [clove.Event](#) class. It implements an event, which can be triggered from one party in certain situations and can be listened to by other parties. Events can be arbitrary things, but often have to do with user interaction like a mouse click on a certain widget.

Although not technically, there is a conceptional distinction between the owner of an event (could be a certain instance of [clove.Button](#)) and the listeners (the program components which want to react on this event). They use a [clove.Event](#) instance in different ways.

Typically event names begin with 'On' like in `OnClicked`.

1.11.1 Handling An Event

Any party can listen to a certain event instance in order to react on it. It needs to have access to the event instance and to call [clove.Event.addHandler](#) on it:

```
mybutton.OnClicked.addHandler(function(e) {  
    // do something  
});
```

The `e` parameter is a [clove.EventArgs](#) instance, which provides some execution control and holds event-specific information.

Alternatively to this way, you can directly specify an event handler in the widget configuration:

```
clove.build({view: "Button", OnClicked: function(e){...}});
```

Please **Note**: Call [clove.Event.removeHandler](#) for unregistering the handler (or see [clove.Event.addHandler](#) for other ways).

1.11.2 Triggering An Event

The owner of an event triggers it when a particular situation occurs. At first it has to construct it and make it available to potential listeners somehow.

```
this.OnSomethingGreatHappened = new clove.Event();
```

Note that there is one instance per owner and that the coupling between them is loose.

The owner triggers the event with some additional information, which leads to the execution of all handlers:

```
this.OnSomethingGreatHappened.trigger({answer: 42});
```

1.12 Datasources

Datasources are an abstraction for how to get and modify a certain data store (either a real one or a virtual one computing live values). Mostly for structured views - as tables, trees, ... - they provide the actual sources of data for displaying.

```
var mydatasource = new MyLottoNumberPredictionDatasource();  
clove.build({view: "TableView", datasource: mydatasource});
```

The abstract base class for each datasource is [clove.Datasource](#).

Before going deeper into the model, a ready-to-use implementation is introduced.

See also the demos about datasources.

1.12.1 Native JavaScript Datasources

While the datasource model is flexible enough for adapting it to arbitrary sources, there is one very simple implementation of it, which can directly be used and populated from outside. Just construct a [clove.NativeDatasource](#) instance and fill it with some data, then assign it to some view.

```
var mydatasource = new clove.NativeDatasource();
mydatasource.root().addRow(["Onion Marmelade", "10€"]);
mydatasource.root().addRow(["Rat Juice", "6.50€"]);
clove.build({view: "TableView", datasource: mydatasource});
```

The interface provides some more. Please find more details in the Developer Documentation.

1.12.2 Datasources Model

As mentioned before, [clove.NativeDatasource](#) is just one implementation of the base class [clove.Datasource](#). You can implement custom subclasses and use them instead. This allows you to take the data not just from a JavaScript object, but from virtually everywhere.

In general, the data model is the following:

- There is one root node.
- Each node has a table like form with rows and columns building cells.
- Each cell has a data value; typically a String or number.
- Each cell has also is a new node.

The first three points are exactly what you would see in the spreadsheet tool of your favorite office suite, at least if you don't think too deeply what 'root node' should mean (maybe the root node is the worksheet in that analogy).

The last point might make it more difficult to imagine this construct graphically. Each cell in this table does not only have a value, but also is one more table. This can be nested arbitrarily deep.

Please **Note**: Every single node in this structure also is a table (it just could be an empty one), while each table cell is a node! Thinking ahead, this means 'node', 'table' and 'table cell' are interchangeable; which can be a confusing thing at first.

Typically you will not need the full flexibility of this model. However, the typical figures can easily be represented in such a model:

- Scalar values: Store your value in the root node.
- Lists: Store your values in the first column of the root node.
- Tables: Store your values in the rows and columns of the root node.
- Trees: Store each first-level nodes in the first column of the root node (as for lists). Then, for each of those nodes, go to the associated cell and insert the direct childs in the same way in its subtable. Do this recursively for each node until the tree is complete.

The [clove.Datasource](#) base class provides an interface which directly reflects the data structure. This must be implemented by a custom subclass, so external data consumers are able to retrieve your data structure by it. The Developer Documentation gives a full interface overview, while the following explains how a datasource works:

- `clove.Datasource.getValue(ptr)`: Returns the value in a certain cell specified somehow by `ptr`.

For the root node, a consumer uses `undefined` as `ptr`. How any inner node, it uses another datasource function to get a pointer to it:

- `clove.Datasource.valuePointer(irow, icol, parent)`: Returns a [clove.DatasourceValuePointer](#), pointing to a non-root node inside the datasource. It is specified by a row and column index and a parent node. If the parent node is not the root node, you would use the same method for getting a pointer to it. This nested calls directly reflect the nested data structure.

Example:

```
mydatasource.getValue(
    mydatasource.valuePointer(1, 3,
        mydatasource.valuePointer(3, 7)
    )
);
```

This returns the value of cell 1/3 of the node in cell 3/7 of the root node.

1.12.2.1 Implementing A Custom Datasource

The already mentioned methods and some more must be implemented in a datasource. The first steps are again about the [clove.DatasourceValuePointer](#) class.

- `clove.Datasource.valuePointer(irow, icol, parent)`: This method constructs and returns such a pointer:

```
valuePointer(irow, icol, parent) {
    //...
    return new clove.DatasourceValuePointer(irow,
        icol, backendObject);
}
```

The major trick here is the `backendObject`. A backend object for a node is an object, which is opaque for Clove and for consumers, but which contains everything your custom datasource implementation needs to answer data queries to that node. This could be custom index structures, references, or direct parts of your custom data structure.

When this function is called with `parent:=undefined`, you have to use an internal representation of the `irow/icol` cell in the root as `backendObject`. Otherwise you have to use the representation of an inner structure. For finding the right path in your structure, you must take care of the backend object you can get from `parent`:

```
var myRootFoo = new Foo(); // our external source
//...

class MyDatasource extends clove.Datasource(Object) {

    valuePointer(irow, icol, parent) {
        if (parent)
            var parentfoo = parent.backendObject;
        else
            var parentfoo = myRootFoo;
        return new clove.DatasourceValuePointer(irow, icol, parentfoo.
            getInnerFoo(irow, icol));
    }

    //...
```

- `clove.Datasource.getValue(ptr)`: Return the value for the specified node:

```
getValue(ptr) {
    return ptr.backendObject.getFooValue();
}
```

This alone should be enough to fetch data from an arbitrary place in your structure. There are some more parts required though:

- `clove.Datasource.rowCount(parent)` and `clove.Datasource.columnCount(parent)`: For a given parent node, return how much rows and columns it has:

```
rowCount(parent) {
    return parent.backendObject.getInnerFooRowCount();
}

// same for columnCount
```

- `clove.Datasource.parent(ptr)`: Return the parent for the node given as [clove.DatasourceValuePointer](#) `ptr`:

```
parent(ptr) {
    var pfoo = ptr.backendObject.getParentFoo();
    if (pfoo == myRootFoo)
        return undefined;
    else
        return new clove.DatasourceValuePointer(pfoo.rowindex(), pfoo.
            columnindex(), pfoo);
}
```

The last essential thing is to notify consumers whenever something changed in your data structure. Otherwise you would never - or only randomly - get updated views. It could also fail when it assumes some row or column counts while some of them disappeared meanwhile in your structure.

In order to correctly notify consumers, some events from your [clove.Datasource](#) instance must be triggered:

```
this.OnDataInsert.trigger({r1: ..., r2: ..., c1: ..., c2: ...,
    parent: ...});

this.OnDataRemove.trigger(/* as above */);

this.OnDataUpdate.trigger(/* as above */);
```

Those events are for creation, removal and updating of nodes. The parameters are `parent`, which is a pointer as described above, and first/last row and column index which is affected by the event.

1.12.3 Headersources

There is an additional concept which is about the headers of rows and columns. In order to configure them (mostly the header text), some views also accept a headersource. This is a different interface, potentially implemented by a different object, which provides those information for entire rows and columns. However, the interface works similar and very often both interface are implemented by the same object.

Please read about [clove.Headersource](#) for details.

Please **Note**: The [clove.NativeDatasource](#) implementation introduced above also implements this interface and also provides methods for configuring the headers from outside.

1.12.4 Data Bindings

While datasources are the only way to display data in [clove.DataView](#) widgets, they are also part of the powerful data binding foundation. This allows to bind a datasource to arbitrary properties of arbitrary widgets.

For binding a node in a [clove.Datasource](#) to a property of a widget, a [clove.DataBinding](#) is to be used. This should be created with the [clove.databind](#) function and

- either connected in a widget configuration:

```
clove.build({view: "EditBox", text: clove.databind({
  datasource: mydatasource})});
```

- or later on with [clove.Widget.bindProperty](#):

```
mywidget.bindProperty("text", clove.databind({datasource: mydatasource}));
```

Read about [clove.databind](#) for more options.

1.12.5 Asynchronous Data Sources

Asynchronous data sources are implemented by [clove.AsyncDatasource](#). There are some interesting subclasses like [clove.AjaxAsyncDatasource](#). Read the api documentation for details.

1.13 Dialogs

A dialog is a body area with a title bar, floating above the rest of the interface (looks and feels like dialogs in desktop environments). Be aware that they are simulations which run inside the main browser window. Opening an actual popup or a new browser tab is a different story.

There are some ways to create dialogs, with different simplicity levels which of course also differ in power.

See also the *HeinersAsiaShop* demo.

1.13.1 Simple Dialogs

Message dialogs with [clove.messageDialog](#) can show a message text in a dialog and requests the user to press a button. This dialog is modal, so the interface behind is not reachable. When the user has clicked on a button, the dialog is closed and an event is triggered. Example:

```
var OnProceed = clove.messageDialog({
  message: "Hello World!",
  buttons: ["Foo", "Bar", "Baz"],
});
OnProceed.addHandler(function(e) {
  console.log("Clicked on #" + e.button);
});
```

A similar dialog with a input text box is provided by [clove.inputDialog](#). A slightly more complicated variant with a custom inner part (but including the button bar) is provided by [clove.conversationDialog](#). Read about them for details.

1.13.2 Complex Dialogs

The widget class `clove.Dialog` implements the container with the frame and title bar, used to be the dialog. Like other containers, it has a property for an inner widget. But the usage is typically different, because direct usage in `clove.build` would lead either to a full-screen dialog or to some dialog-like looking box within your main interface. Both is obviously not intended behavior for a dialog.

Instead, use `clove.Dialog.show`, mostly in the same way as `clove.build`, and build an arbitrary inner interface for this dialog:

```
var dlg = clove.Dialog.show({
  view: "Dialog",
  title: "My Dialog",
  body: {view: "Label", label: "Hello World!"},
});
```

Although not for `clove.build`, the return value of `clove.Dialog.show` is important to keep. Everything you build with this function is not part of the root `namespace`. Instead, you must use `dlg.namespace` for getting widgets by name.

The `clove.Dialog` widget itself is also available in this result as `dlg.widget`. This is particularly useful for eventually closing the dialog later on with `dlg.widget.close()`.

1.13.3 Popups

Even more generic is to use `clove.utils.popup`. This opens any kind of widget floating in the same way as a dialog would do. Read about it for details.

1.14 Internationalization

Internationalization of a user interface has many facets. Some of them should be handled fine directly by the browser (this is true e.g. for string representations for numbers, dates and times). String translation is the next big topic and is what Clove helps you with.

There is even more, like left-to-right vs. right-to-left text flow. But those are beyond the scope of this manual.

The Clove Internationalization support is implemented in the `clove.i18n` class. There exists a single instance of this class as `clove.i18n`.

The idea behind translating a user interface is simple:

- Collect all your interface strings in a table, translated into each target language:

```
clove.i18n.addString('HelloWorld', {en:"Hello world", de:"Hallo Welt", nl:"Hallo wereld"});
clove.i18n.addString('ThanksForVisiting', {en:"Thank you for v...});
```

- Take strings from this object whenever a user interface text is build:

```
foolabel.setLabel( clove.i18n.HelloWorld );
```

This automatically does the translation to the language set in the user's system settings.

1.15 Custom Widgets

For complex and dynamic interfaces it might be a good idea to encapsulate some isolated parts to a new custom widget. Another reason for a custom widget could be to implement some entirely new behavior or functionality.

A custom widget class ...

- either directly or indirectly inherits from [clove.Widget](#)
- provides a constructor with the same signature as [clove.Widget](#)
- overrides some of the methods, depending on what it should do
- often introduces new [properties](#)
- virtually always implements adding some content to the html dom tree and working with it

A rather minimalist widget implementation is the following:

```
class MyHelloWorldLabel extends clove.Widget {  
  
    constructor(config, domnode) {  
        super(config, domnode);  
        // this.contentnode is our html dom node  
        this.contentnode.textContent = "Hello World";  
    }  
  
    // optional  
    doinit() {  
        // initialization steps ...  
    }  
}
```

Once this class is defined, you can use it in widget configurations:

```
clove.build({view: "MyHelloWorldLabel"});
```

For all kinds of graphical representation, you should use css as often as you can and especially follow the [Styling](#) section.

The `config` object is the widget configuration like it is passed in at the `clove.build` call. However, on the way down the constructor chain, this object can (and typically will) get changed. [clove.utils.applyDefaultsToConfig](#) is a typical tool in this context.

1.15.1 Widget Properties

Instead of the static text from the example above, you often want to provide a way to let the widget consumer decide about content or behavior. The widget property system, which does exactly this, was already introduced above from the consumer perspective. For the widget developer, providing a property `myproperty` means the following:

- Calling `this.declareProperty("myproperty", mydefaultvalue)` in the constructor. The default value is optional.
- Optionally provide `myproperty_getter` and/or `myproperty_setter` for dynamically computing property data or for reacting on changes, e.g. by changing stuff in the html content.

1.15.2 Layouts And Sizing

Widgets have to live together on a more or less large area, where they get a position and size. With the size actually allocated for the widget, the internal routines then have to align the own content in a proper way. The custom widget has to play together with these mechanisms. This can happen more or less complicated, as the following sections describe.

But beforehand, please note: `clove.Widget` comes with an existing implementation for all this sizing matters, which works for some simpler cases of html content. It will however fail whenever the content does some internal positioning or uses block elements and in many other cases.

1.15.2.1 Using An Existing Layout Implementation

A very easy and powerful way to build new widgets is to compose it of existing widgets and use a layout internally, instead of manually generating new html content. Obviously this does only work if the existing widgets are enough to build your custom one.

Implement such a composed widget by adding one of the layout mixings to the superclass chain and directly give it a configuration:

```
class MyLabelComposition extends clove.StackLayout(clove.Widget) {
  constructor(config, domnode) {
    // add some more stuff to (a copy of) the config
    config = clove.utils.applyDefaultsToConfig(config, {
      orientation: "vertical",
      children: [
        {view: "Label", label: "Foo"},
        {view: "Label", label: "Bar"},
      ],
    });
    super(config, domnode);
  }
}
```

1.15.2.2 Provide Custom Sizing Support

If the custom widget manages real own html content, it typically has to implement the following methods for sizing support:

- `clove.Widget.computeMinimalWidth`, `clove.Widget.computePreferredWidth`, `clove.Widget.computeMinimalHeightForWidth`, `clove.Widget.computePreferredHeightForWidth`: Return minimal and preferred sizes on both axes for the widget in its current state as pixel values.
- `clove.Widget.doresize`: Realign the internal stuff properly into the current actual widget size.

Whenever the some relevant parts of the widget state changed, it has to call `clove.Widget.relayout` in order to refresh its sizes.

1.15.2.3 Prevent Name Conflicts

When you implement a new custom widget by somehow composing it of existing ones, you typically assign names to some internal parts, so you can access and work with them internally later on. But there is a pitfall: Those names would conflict with other ones outside of the widgets, or even with a second instance of the same custom widget. This can be solved this way:

- Place your custom widget in a new namespace. This can be done by adding `clove.NameScope` to your inheritance chain, so you might end up with a superclass like `clove.NameScope(clove.StackLayout(clove.Widget))`. This leads to an isolated namespace for all your inner widgets.
- If your container isn't a container, you are done. Otherwise there is a new problem now: When a consumer uses the new custom widget, placing also widgets in this container, and tries to access those ones by name, it will fail. This is because those widgets aren't in the namespace you would expect, but they are in that isolated new one. The solution is to introduce one more namespace, just for this containing part, and add just this one as child namespace to the original one.

```
class MyContainerWidget extends clove.NameScope(clove.StackLayout(clove.Widget)) {
  constructor(config, domnode) {
    super(clove.utils.applyDefaultsToConfig(config, {
      children: [
        ...,
        {rows: [], name: "innercontainer", newNameScope: true},
      ],
    ), domnode);
    this.declareProperty("body", {rows: []});
  }

  body_setter(v) {
    this.getByName("innercontainer").setChildren([v]);
  }

  doinit() {
    super.doinit.call(this);
    this.containingNameScope().addChildNameScope(this.getByName("scroll"));
  }
}
```

1.15.3 Best Practices

There are some guidelines for developing custom widgets, which should be known.

Add a css class to the top node

There is a another html dom node in each widget, `this.topnode`, which encloses the actual content node. This has different technical purposes and is also typically used to assign a css class to, which represents the widget class. This allows to style the widget parts.

```
class MyFooWidget extends clove.Widget {
  constructor(config, domnode) {
    super(config, domnode);
    $(this.topnode).addClass("myfoowidget");
    //...
  }
}
```

clove.utils.suspendResizing

When a program logic does large changes on some user interface parts, this can be time consuming. Think about adding a large bunch of data to a widget. Depending on how it is designed, you might end up with calling something like `somewidget.addData(something)`; lots of times. Each of this call likely will trigger the recomputation of the widget sizing. Most of those computations - all but the last to be exact - have no value at all, but can take a considerable amount of time.

For large computations within your widget routines, you should temporarily suspend the resizing.

```
var xxx = clove.utils.suspendResizing();
try {
    //...
}
finally {
    clove.utils.resumeResizing(xxx);
}
```

1.16 Styling

Styling in Clove should be done entirely with css classes. There are only rare cases where directly assigning styles to elements is the better way. Since a Clove widget should be represented by a css class as well, styling is natural.

```
.myfoowidget {
    color: blue;
}
```

More classes can be added at some place of the dom node for supporting finer styles.

Also note the dom structure each widget has: `parent > topnode > contentnode > content`. The `topnode` is the root node of a widget. Within it, there is the `contentnode` and within it the actual content. For some css selectors this is important to know.

```
.myfoowidget > div > a {
    color: blue;
}
```

This would colour each `a` node directly in the content node of a `MyFooWidget`.

1.16.1 Clove Common Styles

There are some common style classes. They help for easily styling some stuff in default situations, like larger control appearance, important texts, errors texts or margins. It's not required to use them, but they avoid some typing and makes application styling easier to adapt.

For a list of existing common classes, inspect code samples or `clove.css` and find css class names beginning with `clovestyle_`. Those kinds of classes are particularly interesting:

- Text styling: `clovestyle_text_*`
- Panels (i.e. container areas for some content): `clovestyle_panel_*`
- Margins around arbitrary widgets: `clovestyle_*margin*`

1.17 Using Widgets In Existing Webpages

The typical way is to build widgets completely in a Clove context. However, it is also possible to build widgets into existing parts of an existing webpage. This way reuses your existing page but just adds some inner parts to it.

At first you have to get the html dom node which shall be populated by a widget. Then, use this in the build configuration of the `clove.build` call:

```
var mydiv = ...;
clove.build({
  rows: ...
}, {
  domnode: mydiv,
});
```

You might also set the `clove.Widget.doStandaloneResizing` property on such a widget in order to make it automatically resize itself.

1.18 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

clove	177
clove::DataBinding	187
clove::DataBindingConverter	188
clove::Datasource	189
clove::AsyncDatasource	54
clove::AjaxAsyncDatasource	44
clove::NativeDatasource	716
clove::ProxyDatasource	848
clove::FilterProxyDatasource	371
clove::SortProxyDatasource	1012
clove::DatasourceValuePointer	193
clove::Event	346
clove::EventArgs	348
clove::Headersource	453
clove::AjaxAsyncDatasource	44
clove::AsyncDatasource	54
clove::NativeDatasource	716
clove::ProxyDatasource	848
clove::I18N	499
clove::Icon	500
clove::Layout	564
clove::FlatLayout	380
clove::FlatView	385
clove::GridLayout	450
clove::Grid	429
clove::StackLayout	1040
clove::HorizontalStack	456
clove::VerticalStack	1187
clove::WrapLayout	1279
clove::Wrap	1258
clove::NameScope	714
clove::RootNameScope	966

clove::NativeDataSourceNode	727
clove::NotificationController	732
clove::RadioGroup	877
clove::symbols	1043
clove::utils	1176
clove::Widget	1239
clove::AbstractMenu	23
clove::Menubar	650
clove::PopupMenu	782
clove::Border	90
clove::Button	110
clove::PopupMenuButton	804
clove::CheckButton	157
clove::DataView	197
clove::ListView	565
clove::TableView	1044
clove::TreeView	1145
clove::Dialog	253
clove::EditBox	299
clove::DateBox	229
clove::EditComboBox	322
clove::DropDownBox	275
clove::MultilineEditBox	691
clove::NumericEditBox	733
clove::PasswordEditBox	759
clove::TimeBox	1099
clove::Expander	349
clove::FlatView	385
clove::Form	409
clove::Grid	429
clove::HorizontalStack	456
clove::HtmlView	479
clove::IconView	502
clove::ImageView	522
clove::Label	543
clove::MainView	597
clove::MediaPlayer	623
clove::AudioPlayer	63
clove::VideoPlayer	1211
clove::ModalPanel	670
clove::ProgressBar	826
clove::RadioButton	855
clove::RawResizeSplitter	880
clove::ResizeSplitter	900
clove::RichEdit	924
clove::RichEditBox	944
clove::ScrollView	967
clove::Slider	989
clove::Spacer	1021
clove::TabView	1075
clove::Carousel	131
clove::Toolbar	1123
clove::VerticalStack	1187
clove::Wrap	1258

1.19 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

clove::AbstractMenu	A base class for widgets which present a menu of actions to the user	23
clove::AjaxAsyncDatasource	A clove::AsyncDatasource implementation which uses Ajax requests for data retrieval and modification	44
clove::AsyncDatasource	A clove::Datasource implementation which supports asynchronous data operations like Ajax requests	54
clove::AudioPlayer	A widget which plays an audio source and optionally allows user controls	63
clove::Border	A single-widget container framing the inner one with (a css-styled) border	90
clove::Button	A button allows the user to trigger a particular program event	110
clove::Carousel	A carousel container	131
clove::CheckButton	A check button	157
clove	Clove root namespace	177
clove::DataBinding	A data binding	187
clove::DataBindingConverter	A data binding converter maps values between ui representation and data model in a custom way	188
clove::Datasource	Base class for datasources	189
clove::DatasourceValuePointer	A pointer to one node in a clove::Datasource	193
clove::DataView	Base class for some views which shows hierarchical data from a clove::Datasource	197
clove::DateBox	A user input box for date input	229
clove::Dialog	A dialog is a container for a new sub user interface, decorated with a border and a title bar . . .	253
clove::DropdownBox	A single-line text box (without edit functionality) with a popup list of values to select from	275
clove::EditBox	A single-line box for text input from the user	299
clove::EditComboBox	A single-line box for text input from the user with an additional popup list of value recommendations	322
clove::Event	An event. It can have some handlers (or: listeners) and can be triggered	346
clove::EventArgs	Arguments for a particular incidence of a clove::Event	348
clove::Expander	A single-widget container which can be expanded or collapsed	349
clove::FilterProxyDatasource	A clove::Datasource implementation which filters another clove::Datasource	371
clove::FlatLayout	A mixin for flat layout implementations	380
clove::FlatView	A container which shows one of its child widgets in full size and hides all others	385

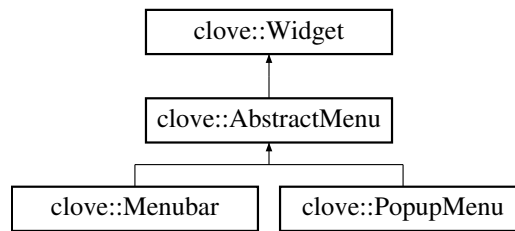
clove::Form	A container for a form-like with a label for each entry in a row-oriented alignment	409
clove::Grid	A container for aligning child widgets in a grid	429
clove::GridLayout	A mixin for grid layout implementations	450
clove::Headersource	Base class for headersources	453
clove::HorizontalStack	A container which stacks child widgets column-wise	456
clove::HtmlView	A host for arbitrary html content	479
clove::I18N	Internationalization support class	499
clove::Icon	An icon	500
clove::IconView	Shows an icon	502
clove::ImageView	Shows an image	522
clove::Label	A text label	543
clove::Layout	A mixin for a widget classes which align child widgets in some way	564
clove::ListView	A view which shows a clove::Datasource in a list	565
clove::MainView	A typical main view, including a clove::Toolbar and a container for an inner body widget	597
clove::MediaPlayer	A base class for media player widgets	623
clove::Menubar	A menu bar	650
clove::ModalPanel	A surface for implementing modality	670
clove::MultilineEditBox	A multi-line box for text input from the user	691
clove::NameScope	A mixin realizing a new namespace	714
clove::NativeDatasource	A clove::Datasource implementation which stores all data in-memory	716
clove::NativeDatasourceNode	A node value implementation for clove::NativeDatasource	727
clove::NotificationController	Controller for clove notifications	732
clove::NumericEditBox	A text box with up/down buttons for numbers	733
clove::PasswordEditBox	A box for password input from the user	759
clove::PopupMenu	A popup menu (i.e. a box with vertically listed actions)	782
clove::PopupMenuButton	A special clove::Button which opens a popup menu when it is triggered	804
clove::ProgressBar	A progress bar	826
clove::ProxyDatasource	A clove::Datasource implementation which proxies the content of another datasource in a some-how transformed way	848

clove::RadioButton	A radio button	855
clove::RadioGroup	Groups some clove::RadioButton together in a logical way, so the user can select exactly one of them	877
clove::RawResizeSplitter	The actual splitter in a clove::ResizeSplitter	880
clove::ResizeSplitter	A container for multiple widgets with splitters for manual resizing between them	900
clove::RichEdit	A user input box for text with rich formatting	924
clove::RichEditBox	A user input box for text with rich formatting	944
clove::RootNameScope	A standalone namescope	966
clove::ScrollView	A container for making the children scrollable	967
clove::Slider	A slider allows the user to choose a number from a given value interval	989
clove::SortProxyDatasource	A clove::Datasource implementation which sorts another clove::Datasource	1012
clove::Spacer	A spacer widget for filling gaps in a layout	1021
clove::StackLayout	A mixin for stack layout implementations	1040
clove::symbols	Namespace for symbol constants	1043
clove::TableView	A view which shows a clove::Datasource in a table	1044
clove::TabView	A tabbed container	1075
clove::TimeBox	A user input box for time input	1099
clove::Toolbar	A toolbar has header texts and a menu bar in a perceptible visual bar, typically used on top of a user interface with all available width	1123
clove::TreeView	A view which shows a clove::Datasource in a tree	1145
clove::utils	Namespace for some utilities	1176
clove::VerticalStack	A container which stacks child widgets row-wise	1187
clove::VideoPlayer	A widget which plays a video source and optionally allows user controls	1211
clove::Widget	Base class for a Clove widget	1239
clove::Wrap	A container for aligning child widgets in horizontal wrapping lines	1258
clove::WrapLayout	A mixin for wrap layout implementations	1279

1.20 `clove::AbstractMenu` Class Reference

A base class for widgets which present a menu of actions to the user.

Inheritance diagram for `clove::AbstractMenu`:



Public Member Functions

- [actions](#) ()
The actions to be shown in this menu.
- [setActions](#) (value)
Setter for [actions\(\)](#).
- [OnActionTriggered](#)
Triggered when a menu action was chosen by the user for execution.
- [OnBeforeSubactionsExpanded](#)
Triggered just before a branch of subaction is expanded.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.

- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (clss)
Adds the css class `clss` to this widget.
- [removeStyleClass](#) (clss)
Removes the css class `clss` from this widget.
- [isStyleClass](#) (clss)
Checks if this widget contains the css class `clss`.
- [setStyleClassAssigned](#) (clss, assigned)
Sets or unsets the css class `clss` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()

- If the widget handles to resize itself as needed.*

 - `setDoStandaloneResizing` (value)
 Setter for `doStandaloneResizing()`.
- `mayFocus` ()
If the widget can have the keyboard focus.
- `setMayFocus` (value)
 Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity` (value)
 Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity` (value)
 Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing` ()
If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing` (value)
 Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing` ()
If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing` (value)
 Setter for `strictVerticalSizing()`.
- `vstretch` ()
Alias for `verticalStretchAffinity()`.
- `setVstretch` (value)
 Setter for `vstretch()`.
- `hstretch` ()
Alias for `horizontalStretchAffinity()`.
- `setHstretch` (value)
 Setter for `hstretch()`.
- `busy` ()
If the widget is in busy state, typically resulting in a loading animation.
- `setBusy` (value)
 Setter for `busy()`.
- `registerBusy` ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy` (token)
Drops a token and returns to normal state if no other tokens exist.
- `hasFocus` ()
If the widget has the keyboard focus.
- `innerSize` ()
Returns inner width and height of this widget.
- `outerSize` ()
Returns outer width and height of this widget.
- `OnDestroyed`
Triggered when this widget was removed somehow.
- `OnResized`
Triggered when this widget was resized.
- `OnVisibilityChanged`
Triggered when the visibility of this widget changed.

- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.20.1 Detailed Description

A base class for widgets which present a menu of actions to the user.

1.20.2 Member Function Documentation

1.20.2.1 actions()

```
actions ( )
```

The actions to be shown in this menu.

It is a list of action configuration objects, each having a structure like `{name:'myaction', label:'My menu action'}`. It can have a list of sub-items in the `subactions` property.

Instead of a simple list, it may also be a [clove::Datasource](#) with the action configuration structures aligned in rows.

Use [clove::MenuSeparator](#) for a separator.

One action allows this properties:

- `name`: The action name.
- `label`: The label string.
- `icon`: A [clove::Icon](#).
- `disabled`: If it is disabled.
- `invisible`: If it is invisible.
- `checkable`: If it is checkable.
- `checked`: If it is checked.

1.20.2.2 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.20.2.3 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.20.2.4 `busy()`

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.20.2.5 `childrenWidgets()`

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.20.2.6 `computeMinimalHeightForWidth()`

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.20.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.20.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.20.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.20.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.20.2.11 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.20.2.12 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.20.2.13 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.20.2.14 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.20.2.15 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.20.2.16 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.20.2.17 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.20.2.18 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.20.2.19 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.20.2.20 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.20.2.21 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.20.2.22 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.20.2.23 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.20.2.24 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.20.2.25 hasFocus()

`hasFocus () [inherited]`

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.20.2.26 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.20.2.27 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.20.2.28 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.20.2.29 `innerSize()`

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.20.2.30 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.20.2.31 `isStyleClass()`

```
isStyleClass (
    css ) [inherited]
```

Checks if this widget contains the css class `css`.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.20.2.32 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.20.2.33 `name()`

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.20.2.34 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.20.2.35 OnActionTriggered()

`OnActionTriggered`

Triggered when a menu action was chosen by the user for execution.

The event arguments contain the selected action name in `action`.

This is a [clove.Event](#) instance. See Manual for details.

1.20.2.36 OnBeforeSubactionsExpanded()

`OnBeforeSubactionsExpanded`

Triggered just before a branch of subaction is expanded.

Implement this event for populating it dynamically.

This is a [clove.Event](#) instance. See Manual for details.

1.20.2.37 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.20.2.38 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.20.2.39 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.20.2.40 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.20.2.41 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.20.2.42 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.20.2.43 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.20.2.44 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.20.2.45 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.20.2.46 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.20.2.47 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.20.2.48 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.20.2.49 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.20.2.50 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.20.2.51 `setActions()`

```
setActions (
    value )
```

Setter for [actions\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.52 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.53 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.54 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.55 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.56 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.57 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.58 setName()

```
setName (  
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.59 setProperty()

```
setProperty (  
    k,  
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.20.2.60 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (  
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.61 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.62 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.63 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.64 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.20.2.65 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (  
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.66 `setVisibility()`

```
setVisibility (  
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.67 `setVstretch()`

```
setVstretch (  
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.20.2.68 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.20.2.69 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.20.2.70 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.20.2.71 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.20.2.72 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.20.2.73 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.20.2.74 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.20.2.75 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

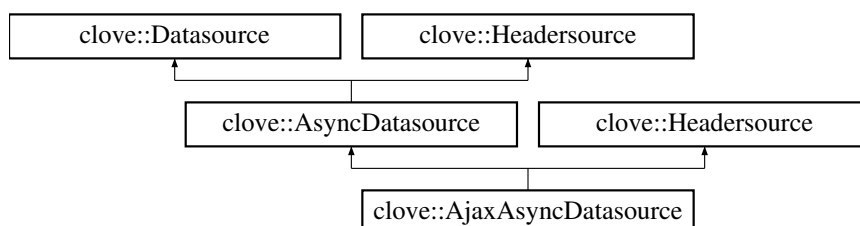
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.21 clove::AjaxAsyncDatasource Class Reference

A [clove::AsyncDatasource](#) implementation which uses Ajax requests for data retrieval and modification.

Inheritance diagram for `clove::AjaxAsyncDatasource`:



Public Member Functions

- [AjaxAsyncDatasource](#) (config)
- [OnDataArrived](#)
Triggered when new data arrived from the server.
- [getAjaxId](#) (ptr)
Returns the Ajax id for a data cell.
- [do_async_pull](#) (ptr, requestConfig)
Executes an asynchronous data modification.
- [do_async_push](#) (ptr, v, requestConfig)
Executes an asynchronous data retrieval.
- [async_pull](#) (ptr, requestConfig)
Asynchronously modifies a data value.
- [async_push](#) (ptr, v, requestConfig)
Asynchronously retrieves a data value.
- [clearCache](#) ()
Clears the internal caches.
- [getValue](#) (ptr)
Returns the value for a given node.
- [getMetadata](#) (ptr)
Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.
- [changeValue](#) (ptr, value)
Change the value for a given node.
- [rowCount](#) (parent)
Returns the number of rows for a given node.
- [columnCount](#) (parent)
Returns the number of columns for a given node.
- [valuePointer](#) (irow, icol, parent)
Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.
- [parent](#) (ptr)
Returns the parent node for a given node.
- [valuePointerNavigateInDepth](#) (ptr, direction, mayexpandfct)
Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.
- [OnDataInsert](#)
Triggered when a node insertion takes place.
- [OnDataRemove](#)
Triggered when a node removal takes place.
- [OnDataUpdate](#)
Triggered when a node data update takes place.
- [getRowHeader](#) (irow, parent)
Returns the header configuration for a given row.
- [getColumnHeader](#) (irow, parent)
Returns the header configuration for a given column.
- [rowHeadersVisible](#) (parent)
If row headers are visible.
- [columnHeadersVisible](#) (parent)
If column headers are visible.
- [OnHeaderDataInsert](#)
Triggered when a new row or column of header data was inserted.
- [OnHeaderDataRemove](#)
Triggered when a row or column of header data was removed.

- [OnHeaderDataUpdate](#)
Triggered when header data were updated.
- [OnHeaderVisibilityUpdated](#)
Triggered when header visibilities changed.
- [OnHeaderDataInsert](#)
Triggered when a new row or column of header data was inserted.
- [OnHeaderDataRemove](#)
Triggered when a row or column of header data was removed.
- [OnHeaderDataUpdate](#)
Triggered when header data were updated.
- [OnHeaderVisibilityUpdated](#)
Triggered when header visibilities changed.

1.21.1 Detailed Description

A [clove::AsyncDatasource](#) implementation which uses Ajax requests for data retrieval and modification.

1.21.2 Constructor & Destructor Documentation

1.21.2.1 AjaxAsyncDatasource()

```
AjaxAsyncDatasource (
    config )
```

The ajax configuration may contain:

- `data_pullUrl`: The url for data retrieval.
- `data_pushUrl`: The url for data modification.
- `data_pullParams`: Additional parameters for data retrieval.
- `data_pushParams`: Additional parameters for data modification.
- `data_params`: Additional parameters.
- `data_mayHaveChildren`: If the data items might have children.
- `data_autoPullCharily`: If automatic data fetching should be less aggressive.
- `params`: Additional parameters.
- `rootType`: The root type. This controls the prefix of the url and parameter keys to consider
- `ajaxAdditionalArgs`: Additional arguments for the ajax request.
- `answerIdKeyName`: The key name for the identification column used in reading server answers.
- `requestIdKeyName`: The key name for the identification used in requests.
- `headerConfigs`: Additional static header configurations. for fetching the first (and often only) level of objects. Instead of `data_`, other prefixes are allowed as well. The return type name of an retrieved object controls which prefix to use for fetching child objects.

Parameters

<i>config</i>	The Ajax configuration.
---------------	-------------------------

1.21.3 Member Function Documentation

1.21.3.1 `async_pull()`

```
async_pull (
    ptr,
    requestConfig ) [inherited]
```

Asynchronously modifies a data value.

Do not override this in custom implementations, use [clove::AsyncDatasource::do_async_pull\(\)](#) instead.

`requestConfig` may contain:

- `onsuccess`: A `function()` called when the operation is completed and reflected in this datasource.
- `onfailed`: A `function(error)` called when an error occurred.
- `reason`: An optional custom object which can be used by observers for custom decisions.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>requestConfig</i>	The request configuration.

1.21.3.2 `async_push()`

```
async_push (
    ptr,
    v,
    requestConfig ) [inherited]
```

Asynchronously retrieves a data value.

Do not override this in custom implementations, use [clove::AsyncDatasource::do_async_push\(\)](#) instead.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>v</i>	The new value.
<i>requestConfig</i>	The request configuration. See clove::AsyncDatasource::async_pull() for details.

1.21.3.3 `changeValue()`

```
changeValue (
    ptr,
    value ) [pure virtual], [inherited]
```

Change the value for a given node.

This method is called from external places, e.g. when a user makes changes in a datasource-connected widget.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>value</i>	The new value.

1.21.3.4 `clearCache()`

```
clearCache ( ) [inherited]
```

Clears the internal caches.

1.21.3.5 `columnCount()`

```
columnCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of columns for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.21.3.6 `columnHeadersVisible()`

```
columnHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If column headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.21.3.7 do_async_pull()

```
do_async_pull (
    ptr,
    requestConfig ) [pure virtual], [inherited]
```

Executes an asynchronous data modification.

Do not call this from outside, use [clove::AsyncDatasource::async_pull\(\)](#) instead.

Override this method in custom implementations.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>requestConfig</i>	The request configuration. See clove::AsyncDatasource::async_pull() for details.

1.21.3.8 do_async_push()

```
do_async_push (
    ptr,
    v,
    requestConfig ) [pure virtual], [inherited]
```

Executes an asynchronous data retrieval.

Do not call this from outside, use [clove::AsyncDatasource::async_push\(\)](#) instead.

Override this method in custom implementations.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>v</i>	The new value.
<i>requestConfig</i>	The request configuration. See clove::AsyncDatasource::async_pull() for details.

1.21.3.9 getAjaxId()

```
getAjaxId (
```

```
ptr )
```

Returns the Ajax id for a data cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.21.3.10 getColumnHeader()

```
getColumnHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given column.

Parameters

<i>irow</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.21.3.11 getMetadata()

```
getMetadata (
    ptr ) [pure virtual], [inherited]
```

Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.21.3.12 getRowHeader()

```
getRowHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given row.

Parameters

<i>irow</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.21.3.13 getValue()

```
getValue (
    ptr ) [pure virtual], [inherited]
```

Returns the value for a given node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.21.3.14 OnDataArrived()

```
OnDataArrived
```

Triggered when new data arrived from the server.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.15 OnDataInsert()

```
OnDataInsert [inherited]
```

Triggered when a node insertion takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.16 OnDataRemove()

```
OnDataRemove [inherited]
```

Triggered when a node removal takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.17 OnDataUpdate()

```
OnDataUpdate [inherited]
```

Triggered when a node data update takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.18 OnHeaderDataInsert() [1/2]

OnHeaderDataInsert [inherited]

Triggered when a new row or column of header data was inserted.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.19 OnHeaderDataInsert() [2/2]

OnHeaderDataInsert [inherited]

Triggered when a new row or column of header data was inserted.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.20 OnHeaderDataRemove() [1/2]

OnHeaderDataRemove [inherited]

Triggered when a row or column of header data was removed.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.21 OnHeaderDataRemove() [2/2]

OnHeaderDataRemove [inherited]

Triggered when a row or column of header data was removed.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.22 OnHeaderDataUpdate() [1/2]

OnHeaderDataUpdate [inherited]

Triggered when header data were updated.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.23 OnHeaderDataUpdate() [2/2]

OnHeaderDataUpdate [inherited]

Triggered when header data were updated.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.24 OnHeaderVisibilityUpdated() [1/2]

OnHeaderVisibilityUpdated [inherited]

Triggered when header visibilities changed.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.25 OnHeaderVisibilityUpdated() [2/2]

OnHeaderVisibilityUpdated [inherited]

Triggered when header visibilities changed.

This is a [clove.Event](#) instance. See Manual for details.

1.21.3.26 parent()

```
parent (
    ptr ) [pure virtual], [inherited]
```

Returns the parent node for a given node.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
------------	---

1.21.3.27 rowCount()

```
rowCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of rows for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.21.3.28 rowHeadersVisible()

```
rowHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If row headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.21.3.29 valuePointer()

```
valuePointer (
    irow,
    icol,
    parent ) [pure virtual], [inherited]
```

Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.

Parameters

<i>irow</i>	The row index.
<i>icol</i>	The column index.
<i>parent</i>	The parent as clove::DatasourceValuePointer .

1.21.3.30 valuePointerNavigateInDepth()

```
valuePointerNavigateInDepth (
    ptr,
    direction,
    mayexpandfct ) [inherited]
```

Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
<i>direction</i>	The direction (+1 or -1).
<i>mayexpandfct</i>	A function(<i>ptr</i>) which returns <code>true</code> iff this node's children are to be traversed.

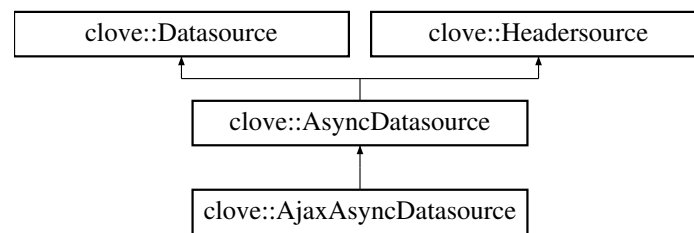
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.22 clove::AsyncDatasource Class Reference

A [clove::Datasource](#) implementation which supports asynchronous data operations like Ajax requests.

Inheritance diagram for clove::AsyncDatasource:



Public Member Functions

- [AsyncDatasource](#) (config)
- [do_async_pull](#) (ptr, requestConfig)
Executes an asynchronous data modification.
- [do_async_push](#) (ptr, v, requestConfig)
Executes an asynchronous data retrieval.
- [async_pull](#) (ptr, requestConfig)
Asynchronously modifies a data value.
- [async_push](#) (ptr, v, requestConfig)
Asynchronously retrieves a data value.
- [clearCache](#) ()
Clears the internal caches.
- [getValue](#) (ptr)
Returns the value for a given node.
- [getMetadata](#) (ptr)
Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.
- [changeValue](#) (ptr, value)
Change the value for a given node.
- [rowCount](#) (parent)
Returns the number of rows for a given node.
- [columnCount](#) (parent)
Returns the number of columns for a given node.
- [valuePointer](#) (irow, icol, parent)
Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.
- [parent](#) (ptr)
Returns the parent node for a given node.
- [valuePointerNavigateInDepth](#) (ptr, direction, mayexpandfct)
Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.
- [OnDataInsert](#)
Triggered when a node insertion takes place.
- [OnDataRemove](#)
Triggered when a node removal takes place.
- [OnDataUpdate](#)
Triggered when a node data update takes place.
- [getRowHeader](#) (irow, parent)
Returns the header configuration for a given row.
- [getColumnHeader](#) (irow, parent)
Returns the header configuration for a given column.
- [rowHeadersVisible](#) (parent)

- If row headers are visible.
 - [columnHeadersVisible](#) (parent)
 - If column headers are visible.
 - [OnHeaderDataInsert](#)
 - Triggered when a new row or column of header data was inserted.
 - [OnHeaderDataRemove](#)
 - Triggered when a row or column of header data was removed.
 - [OnHeaderDataUpdate](#)
 - Triggered when header data were updated.
 - [OnHeaderVisibilityUpdated](#)
 - Triggered when header visibilities changed.

1.22.1 Detailed Description

A [Clove::Datasource](#) implementation which supports asynchronous data operations like Ajax requests.

1.22.2 Constructor & Destructor Documentation

1.22.2.1 AsyncDatasource()

```
AsyncDatasource (
    config )
```

The configuration may contain:

- 'errorhandler': Optional function(error) as fallback error handler.
- 'cacheAge': The maximum cache age in seconds.

Parameters

<code>config</code>	The async datasource configuration.
---------------------	-------------------------------------

1.22.3 Member Function Documentation

1.22.3.1 async_pull()

```
async_pull (
    ptr,
    requestConfig )
```


Asynchronously modifies a data value.

Do not override this in custom implementations, use [clove::AsyncDatasource::do_async_pull\(\)](#) instead.

`requestConfig` may contain:

- `onsuccess`: A `function()` called when the operation is completed and reflected in this datasource.
- `onfailed`: A `function(error)` called when an error occurred.
- `reason`: An optional custom object which can be used by observers for custom decisions.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>requestConfig</i>	The request configuration.

1.22.3.2 `async_push()`

```
async_push (
    ptr,
    v,
    requestConfig )
```

Asynchronously retrieves a data value.

Do not override this in custom implementations, use [clove::AsyncDatasource::do_async_push\(\)](#) instead.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>v</i>	The new value.
<i>requestConfig</i>	The request configuration. See clove::AsyncDatasource::async_pull() for details.

1.22.3.3 `changeValue()`

```
changeValue (
    ptr,
    value ) [pure virtual], [inherited]
```

Change the value for a given node.

This method is called from external places, e.g. when a user makes changes in a datasource-connected widget.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>value</i>	The new value.

1.22.3.4 clearCache()

```
clearCache ( )
```

Clears the internal caches.

1.22.3.5 columnCount()

```
columnCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of columns for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.22.3.6 columnHeadersVisible()

```
columnHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If column headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.22.3.7 do_async_pull()

```
do_async_pull (
    ptr,
    requestConfig ) [pure virtual]
```

Executes an asynchronous data modification.

Do not call this from outside, use [clove::AsyncDatasource::async_pull\(\)](#) instead.

Override this method in custom implementations.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>requestConfig</i>	The request configuration. See clove::AsyncDatasource::async_pull() for details.

1.22.3.8 do_async_push()

```
do_async_push (
    ptr,
    v,
    requestConfig ) [pure virtual]
```

Executes an asynchronous data retrieval.

Do not call this from outside, use [clove::AsyncDatasource::async_push\(\)](#) instead.

Override this method in custom implementations.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>v</i>	The new value.
<i>requestConfig</i>	The request configuration. See clove::AsyncDatasource::async_pull() for details.

1.22.3.9 getColumnHeader()

```
getColumnHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given column.

Parameters

<i>irow</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.22.3.10 getMetadata()

```
getMetadata (
    ptr ) [pure virtual], [inherited]
```

Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.22.3.11 `getRowHeader()`

```
getRowHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given row.

Parameters

<i>irow</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.22.3.12 `getValue()`

```
getValue (
    ptr ) [pure virtual], [inherited]
```

Returns the value for a given node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.22.3.13 `OnDataInsert()`

```
OnDataInsert [inherited]
```

Triggered when a node insertion takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.22.3.14 `OnDataRemove()`

```
OnDataRemove [inherited]
```

Triggered when a node removal takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.22.3.15 OnDataUpdate()

OnDataUpdate [inherited]

Triggered when a node data update takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.22.3.16 OnHeaderDataInsert()

OnHeaderDataInsert [inherited]

Triggered when a new row or column of header data was inserted.

This is a [clove.Event](#) instance. See Manual for details.

1.22.3.17 OnHeaderDataRemove()

OnHeaderDataRemove [inherited]

Triggered when a row or column of header data was removed.

This is a [clove.Event](#) instance. See Manual for details.

1.22.3.18 OnHeaderDataUpdate()

OnHeaderDataUpdate [inherited]

Triggered when header data were updated.

This is a [clove.Event](#) instance. See Manual for details.

1.22.3.19 OnHeaderVisibilityUpdated()

OnHeaderVisibilityUpdated [inherited]

Triggered when header visibilities changed.

This is a [clove.Event](#) instance. See Manual for details.

1.22.3.20 parent()

```
parent (
    ptr ) [pure virtual], [inherited]
```

Returns the parent node for a given node.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
------------	---

1.22.3.21 rowCount()

```
rowCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of rows for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.22.3.22 rowHeadersVisible()

```
rowHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If row headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.22.3.23 valuePointer()

```
valuePointer (
    irow,
    icol,
    parent ) [pure virtual], [inherited]
```

Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.

Parameters

<i>irow</i>	The row index.
<i>icol</i>	The column index.
<i>parent</i>	The parent as clove::DatasourceValuePointer .

1.22.3.24 valuePointerNavigateInDepth()

```
valuePointerNavigateInDepth (
    ptr,
    direction,
    mayexpandfct ) [inherited]
```

Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
<i>direction</i>	The direction (+1 or -1).
<i>mayexpandfct</i>	A function(<i>ptr</i>) which returns <code>true</code> iff this node's children are to be traversed.

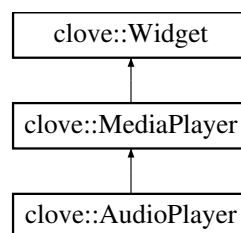
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.23 clove::AudioPlayer Class Reference

A widget which plays an audio source and optionally allows user controls.

Inheritance diagram for `clove::AudioPlayer`:



Public Member Functions

- static [canPlayType](#) (t)
Determines if the browser can play a certain audio type and returns a string as described in the html specs (-> "can<→ PlayType").
- [autoPlay](#) ()
If to automatically start playback as soon as possible.
- [setAutoPlay](#) (value)
Setter for [autoPlay\(\)](#).
- [showControls](#) ()
If to show user controls for play, pause and more.
- [setShowControls](#) (value)

- Setter for [showControls\(\)](#).
- [currentMediaPosition](#) ()
 - The current media playback position in seconds.
- [setCurrentMediaPosition](#) (value)
 - Setter for [currentMediaPosition\(\)](#).
- [loop](#) ()
 - If to playback in an endless loop.
- [setLoop](#) (value)
 - Setter for [loop\(\)](#).
- [muted](#) ()
 - If to mute the audio playback.
- [setMuted](#) (value)
 - Setter for [muted\(\)](#).
- [preload](#) ()
 - If to begin loading the media data as soon as possible.
- [setPreload](#) (value)
 - Setter for [preload\(\)](#).
- [volume](#) ()
 - The audio playback volume between 0 . 0 and 1 . 0.
- [setVolume](#) (value)
 - Setter for [volume\(\)](#).
- [source](#) ()
 - The media source URL.
- [setSource](#) (value)
 - Setter for [source\(\)](#).
- [duration](#) ()
 - The duration of the loaded media source in seconds.
- [hasEnded](#) ()
 - If the playback has ended (i.e. reached the end).
- [isPaused](#) ()
 - If the playback is logically paused.
- [nativeNode](#) ()
 - Returns the native html dom node.
- [play](#) ()
 - Starts playback.
- [pause](#) ()
 - Pauses playback.
- [OnLoadingAborted](#)
 - Triggered when the media loading aborted for some reasons (e.g. network errors).
- [OnReadyForPlayback](#)
 - Triggered when there are enough data for starting playback.
- [OnDurationChanged](#)
 - Triggered when the playback duration changed.
- [OnHasEnded](#)
 - Triggered when the playback has ended.
- [OnIsPaused](#)
 - Triggered when the playback logically paused.
- [OnIsPlaying](#)
 - Triggered when the playback logically starts.
- [declareProperty](#) (k, defaultV)
 - Declares a widget property.

- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)
Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)
Removes the css class `css` from this widget.
- [isStyleClass](#) (css)
Checks if this widget contains the css class `css`.
- [setStyleClassAssigned](#) (css, assigned)
Sets or unsets the css class `css` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)

- Removes this widget.*

 - `effectivelyEnabled` ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
 - `effectiveVisibility` ()

If this widget and all parent widgets are visible. See also `visibility()`.
 - `childrenWidgets` ()

List of the children `clove::Widget` instances.
 - `parentWidget` ()

The parent `clove::Widget`.
 - `name` ()

The name of the widget.
 - `setName` (value)

Setter for `name()`.
 - `enabled` ()

If this widget is marked as enabled (i.e. can interact with the user).
 - `setEnabled` (value)

Setter for `enabled()`.
 - `styleClass` ()

Custom css class(es).
 - `setStyleClass` (value)

Setter for `styleClass()`.
 - `style` ()

Custom css style string. You should not use that, but `styleClass()` instead.
 - `setStyle` (value)

Setter for `style()`.
 - `visibility` ()

If this widget is visible.
 - `setVisibility` (value)

Setter for `visibility()`.
 - `doStandaloneResizing` ()

If the widget handles to resize itself as needed.
 - `setDoStandaloneResizing` (value)

Setter for `doStandaloneResizing()`.
 - `mayFocus` ()

If the widget can have the keyboard focus.
 - `setMayFocus` (value)

Setter for `mayFocus()`.
 - `horizontalStretchAffinity` ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
 - `setHorizontalStretchAffinity` (value)

Setter for `horizontalStretchAffinity()`.
 - `verticalStretchAffinity` ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
 - `setVerticalStretchAffinity` (value)

Setter for `verticalStretchAffinity()`.
 - `strictHorizontalSizing` ()

If the widget is strictly forbidden to get additional horizontal space.
 - `setStrictHorizontalSizing` (value)

Setter for `strictHorizontalSizing()`.
 - `strictVerticalSizing` ()

If the widget is strictly forbidden to get additional vertical space.

- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.23.1 Detailed Description

A widget which plays an audio source and optionally allows user controls.

1.23.2 Member Function Documentation

1.23.2.1 [addStyleClass\(\)](#)

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.23.2.2 autoPlay()

```
autoPlay ( ) [inherited]
```

If to automatically start playback as soon as possible.

1.23.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.23.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.23.2.5 canPlayType()

```
static canPlayType (
    t )
```

Determines if the browser can play a certain audio type and returns a string as described in the html specs (->"canPlayType").

Parameters

<i>t</i>	A mimetype string.
----------	--------------------

1.23.2.6 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.23.2.7 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.23.2.8 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.23.2.9 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.23.2.10 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.23.2.11 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.23.2.12 currentMediaPosition()

```
currentMediaPosition ( ) [inherited]
```

The current media playback position in seconds.

1.23.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.23.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.23.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.23.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.23.2.17 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.23.2.18 duration()

```
duration ( ) [inherited]
```

The duration of the loaded media source in seconds.

1.23.2.19 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.23.2.20 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.23.2.21 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.23.2.22 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.23.2.23 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.23.2.24 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.23.2.25 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.23.2.26 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.23.2.27 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.23.2.28 hasEnded()

```
hasEnded ( ) [inherited]
```

If the playback has ended (i.e. reached the end).

1.23.2.29 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.23.2.30 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.23.2.31 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.23.2.32 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.23.2.33 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.23.2.34 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.23.2.35 isPaused()

```
isPaused ( ) [inherited]
```

If the playback is logically paused.

This does not return `true` just when loading stalls.

1.23.2.36 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.23.2.37 loop()

```
loop ( ) [inherited]
```

If to playback in an endless loop.

1.23.2.38 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.23.2.39 muted()

`muted () [inherited]`

If to mute the audio playback.

1.23.2.40 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.23.2.41 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.23.2.42 nativeNode()

`nativeNode () [inherited]`

Returns the native html dom node.

1.23.2.43 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.44 OnDurationChanged()

`OnDurationChanged [inherited]`

Triggered when the playback duration changed.

See also [duration\(\)](#).

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.45 OnHasEnded()

OnHasEnded [inherited]

Triggered when the playback has ended.

See also [hasEnded\(\)](#).

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.46 OnIsPaused()

OnIsPaused [inherited]

Triggered when the playback logically paused.

This is not triggered when loading stalls.

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.47 OnIsPlaying()

OnIsPlaying [inherited]

Triggered when the playback logically starts.

This is not triggered when loading has stalled and resumes.

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.48 OnKeyDown()

OnKeyDown [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.49 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.50 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.51 OnLoadingAborted()

OnLoadingAborted [inherited]

Triggered when the media loading aborted for some reasons (e.g. network errors).

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.52 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.53 OnReadyForPlayback()

OnReadyForPlayback [inherited]

Triggered when there are enough data for starting playback.

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.54 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.55 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.23.2.56 `outerSize()`

`outerSize () [inherited]`

Returns outer width and height of this widget.

1.23.2.57 `parentWidget()`

`parentWidget () [inherited]`

The parent [clove::Widget](#).

1.23.2.58 `pause()`

`pause () [inherited]`

Pauses playback.

1.23.2.59 `play()`

`play () [inherited]`

Starts playback.

1.23.2.60 `preload()`

`preload () [inherited]`

If to begin loading the media data as soon as possible.

1.23.2.61 `registerBusy()`

`registerBusy () [inherited]`

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.23.2.62 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.23.2.63 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.23.2.64 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.23.2.65 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.23.2.66 setAutoPlay()

```
setAutoPlay (
    value ) [inherited]
```

Setter for [autoPlay\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.67 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.68 setCurrentMediaPosition()

```
setCurrentMediaPosition (
    value ) [inherited]
```

Setter for [currentMediaPosition\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.69 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.70 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.71 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.72 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.73 setLoop()

```
setLoop (
    value ) [inherited]
```

Setter for [loop\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.74 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.75 setMuted()

```
setMuted (
    value ) [inherited]
```

Setter for [muted\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.76 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.77 setPreload()

```
setPreload (  
    value ) [inherited]
```

Setter for [preload\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.78 setProperty()

```
setProperty (  
    k,  
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.23.2.79 setShowControls()

```
setShowControls (  
    value ) [inherited]
```

Setter for [showControls\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.80 setSource()

```
setSource (
    value ) [inherited]
```

Setter for [source\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.81 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.82 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.83 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.84 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.85 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.23.2.86 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.87 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.88 setVolume()

```
setVolume (
    value ) [inherited]
```

Setter for [volume\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.89 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.23.2.90 showControls()

```
showControls ( ) [inherited]
```

If to show user controls for play, pause and more.

1.23.2.91 source()

```
source ( ) [inherited]
```

The media source URL.

1.23.2.92 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.23.2.93 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.23.2.94 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.23.2.95 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.23.2.96 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.23.2.97 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.23.2.98 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.23.2.99 volume()

`volume () [inherited]`

The audio playback volume between 0.0 and 1.0.

1.23.2.100 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

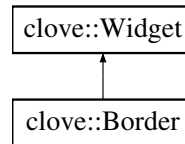
The documentation for this class was generated from the following file:

- `_meta/readme/clove.js`

1.24 clove::Border Class Reference

A single-widget container framing the inner one with (a css-styled) border.

Inheritance diagram for clove::Border:



Public Member Functions

- [body](#) ()
The inner widget as widget configuration like for [clove::build\(\)](#).
- [setBody](#) (value)
Setter for [body\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.

- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (clss)
Adds the css class `clss` to this widget.
- [removeStyleClass](#) (clss)
Removes the css class `clss` from this widget.
- [isStyleClass](#) (clss)
Checks if this widget contains the css class `clss`.
- [setStyleClassAssigned](#) (clss, assigned)
Sets or unsets the css class `clss` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()

- If the widget handles to resize itself as needed.*

 - [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.

- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.24.1 Detailed Description

A single-widget container framing the inner one with (a css-styled) border.

1.24.2 Member Function Documentation

1.24.2.1 `addStyleClass()`

```
addStyleClass (  
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.24.2.2 `bindProperty()`

```
bindProperty (  
    k,  
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<code>k</code>	The widget property name.
<code>vb</code>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.24.2.3 body()

```
body ( )
```

The inner widget as widget configuration like for [clove::build\(\)](#).

1.24.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.24.2.5 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.24.2.6 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.24.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.24.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.24.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.24.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.24.2.11 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.24.2.12 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.24.2.13 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.24.2.14 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.24.2.15 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.24.2.16 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.24.2.17 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.24.2.18 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.24.2.19 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.24.2.20 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.24.2.21 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.24.2.22 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.24.2.23 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.24.2.24 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.24.2.25 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.24.2.26 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.24.2.27 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.24.2.28 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.24.2.29 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.24.2.30 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.24.2.31 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.24.2.32 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.24.2.33 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.24.2.34 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.24.2.35 OnDestroyed()

```
OnDestroyed [inherited]
```

Triggered when this widget was removed somehow.

This is a [Clove.Event](#) instance. See Manual for details.

1.24.2.36 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.24.2.37 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.24.2.38 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.24.2.39 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.24.2.40 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.24.2.41 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.24.2.42 `outerSize()`

```
outerSize ( ) [inherited]
```

Returns outer width and height of this widget.

1.24.2.43 `parentWidget()`

```
parentWidget ( ) [inherited]
```

The parent [Clove::Widget](#).

1.24.2.44 `registerBusy()`

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.24.2.45 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.24.2.46 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.24.2.47 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.24.2.48 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.24.2.49 setBody()

```
setBody (
    value )
```

Setter for [body\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.50 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.51 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.52 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.53 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.54 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.55 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.56 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.57 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.24.2.58 `setStrictHorizontalSizing()`

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.59 `setStrictVerticalSizing()`

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.60 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.61 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.62 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.24.2.63 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.64 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.65 `setVstretch()`

```
setVstretch (
    value ) [inherited]
```

Setter for `vstretch()`.

Parameters

<i>value</i>	The new value.
--------------	----------------

1.24.2.66 `strictHorizontalSizing()`

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.24.2.67 `strictVerticalSizing()`

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.24.2.68 `style()`

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.24.2.69 styleClass()

`styleClass () [inherited]`

Custom css class(es).

1.24.2.70 unregisterBusy()

`unregisterBusy (
 token) [inherited]`

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.24.2.71 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.24.2.72 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.24.2.73 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

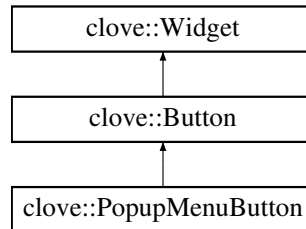
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.25 clove::Button Class Reference

A button allows the user to trigger a particular program event.

Inheritance diagram for clove::Button:



Public Member Functions

- [label](#) ()
The label text.
- [setLabel](#) (value)
Setter for [label\(\)](#).
- [icon](#) ()
The optional [clove::Icon](#) button icon.
- [setIcon](#) (value)
Setter for [icon\(\)](#).
- [checkable](#) ()
If the button is checkable (i.e. has a checked-flag).
- [setCheckable](#) (value)
Setter for [checkable\(\)](#).
- [checked](#) ()
For a checkable button, return if it is checked.
- [setChecked](#) (value)
Setter for [checked\(\)](#).
- [OnClicked](#)
Triggered when the user clicks on this button.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.

- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (class)
Adds the css class `class` to this widget.
- [removeStyleClass](#) (class)
Removes the css class `class` from this widget.
- [isStyleClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children [clove::Widget](#) instances.
- [parentWidget](#) ()
The parent [clove::Widget](#).
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).

- **setEnabled** (value)
Setter for *enabled()*.
- **styleClass** ()
Custom css class(es).
- **setStyleClass** (value)
Setter for *styleClass()*.
- **style** ()
Custom css style string. You should not use that, but *styleClass()* instead.
- **setStyle** (value)
Setter for *style()*.
- **visibility** ()
If this widget is visible.
- **setVisibility** (value)
Setter for *visibility()*.
- **doStandaloneResizing** ()
If the widget handles to resize itself as needed.
- **setDoStandaloneResizing** (value)
Setter for *doStandaloneResizing()*.
- **mayFocus** ()
If the widget can have the keyboard focus.
- **setMayFocus** (value)
Setter for *mayFocus()*.
- **horizontalStretchAffinity** ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- **setHorizontalStretchAffinity** (value)
Setter for *horizontalStretchAffinity()*.
- **verticalStretchAffinity** ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- **setVerticalStretchAffinity** (value)
Setter for *verticalStretchAffinity()*.
- **strictHorizontalSizing** ()
If the widget is strictly forbidden to get additional horizontal space.
- **setStrictHorizontalSizing** (value)
Setter for *strictHorizontalSizing()*.
- **strictVerticalSizing** ()
If the widget is strictly forbidden to get additional vertical space.
- **setStrictVerticalSizing** (value)
Setter for *strictVerticalSizing()*.
- **vstretch** ()
Alias for *verticalStretchAffinity()*.
- **setVstretch** (value)
Setter for *vstretch()*.
- **hstretch** ()
Alias for *horizontalStretchAffinity()*.
- **setHstretch** (value)
Setter for *hstretch()*.
- **busy** ()
If the widget is in busy state, typically resulting in a loading animation.
- **setBusy** (value)
Setter for *busy()*.

- [registerBusy \(\)](#)
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy \(token\)](#)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus \(\)](#)
If the widget has the keyboard focus.
- [innerSize \(\)](#)
Returns inner width and height of this widget.
- [outerSize \(\)](#)
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.25.1 Detailed Description

A button allows the user to trigger a particular program event.

1.25.2 Member Function Documentation

1.25.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.25.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.25.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.25.2.4 checkable()

```
checkable ( )
```

If the button is checkable (i.e. has a checked-flag).

If it has, the checked-flag is controlled by [checked\(\)](#).

1.25.2.5 checked()

```
checked ( )
```

For a checkable button, return if it is checked.

See also [checkable\(\)](#).

User interactions do not toggle the checked flag by default. This must be scripted in own event handlers as needed.

1.25.2.6 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.25.2.7 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.25.2.8 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.25.2.9 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.25.2.10 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.25.2.11 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.25.2.12 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.25.2.13 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.25.2.14 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.25.2.15 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.25.2.16 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.25.2.17 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.25.2.18 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.25.2.19 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.25.2.20 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.25.2.21 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.25.2.22 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.25.2.23 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.25.2.24 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.25.2.25 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.25.2.26 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.25.2.27 `horizontalStretchAffinity()`

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.25.2.28 `hstretch()`

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.25.2.29 `icon()`

```
icon ( )
```

The optional [clove::Icon](#) button icon.

1.25.2.30 `init()`

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.25.2.31 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.25.2.32 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.25.2.33 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.25.2.34 label()

```
label ( )
```

The label text.

1.25.2.35 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.25.2.36 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.25.2.37 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.25.2.38 OnClicked()

`OnClicked`

Triggered when the user clicks on this button.

This is a [clove.Event](#) instance. See Manual for details.

1.25.2.39 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.25.2.40 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.25.2.41 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.25.2.42 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.25.2.43 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.25.2.44 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.25.2.45 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.25.2.46 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.25.2.47 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.25.2.48 `registerBusy()`

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.25.2.49 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.25.2.50 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<code>removeconfig</code>	The removal configuration (optional and only for exotic cases).
---------------------------	---

1.25.2.51 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.25.2.52 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.25.2.53 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.54 `setCheckable()`

```
setCheckable (
    value )
```

Setter for [checkable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.55 `setChecked()`

```
setChecked (
    value )
```

Setter for [checked\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.56 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.57 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.58 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.59 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.60 setIcon()

```
setIcon (  
    value )
```

Setter for [icon\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.61 setLabel()

```
setLabel (  
    value )
```

Setter for [label\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.62 setMayFocus()

```
setMayFocus (  
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.63 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.64 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.25.2.65 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.66 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.67 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.68 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.69 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.25.2.70 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.71 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.72 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.25.2.73 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.25.2.74 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.25.2.75 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.25.2.76 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.25.2.77 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.25.2.78 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.25.2.79 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.25.2.80 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

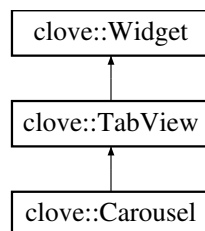
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.26 clove::Carousel Class Reference

A carousel container.

Inheritance diagram for clove::Carousel:



Public Member Functions

- [interval](#) ()
The tab switching interval (in milliseconds).
- [setInterval](#) (value)
Setter for [interval\(\)](#).
- [play](#) ()
Begins automatical tab switching.
- [pause](#) ()
Stops automatic tab switching.
- [isPlaying](#) ()
If automatic tab switching is enabled.
- [next](#) ()
Switches to the next tab.
- [tabs](#) ()

- The list of tabs.*

 - `setTabs` (value)
Setter for `tabs()`.
- `currentTab` ()
The index of the currently selected tab.

 - `setCurrentTab` (value)
Setter for `currentTab()`.
- `userMayAddTabs` ()
If to show a button for adding new tabs.

 - `setUserMayAddTabs` (value)
Setter for `userMayAddTabs()`.
- `tabBarLocation` ()
Where to place the tab bar.

 - `setTabBarLocation` (value)
Setter for `tabBarLocation()`.
- `switchInvisibleAnimationName` ()
Optional animation name for switching away from a tab.

 - `setSwitchInvisibleAnimationName` (value)
Setter for `switchInvisibleAnimationName()`.
- `switchVisibleAnimationName` ()
Optional animation name for switching to a tab.

 - `setSwitchVisibleAnimationName` (value)
Setter for `switchVisibleAnimationName()`.
- `switchInvisibleAnimationDuration` ()
Optional animation duration (in msec) for the switch away animation.

 - `setSwitchInvisibleAnimationDuration` (value)
Setter for `switchInvisibleAnimationDuration()`.
- `switchVisibleAnimationDuration` ()
Optional animation duration (in msec) for the switch animation.

 - `setSwitchVisibleAnimationDuration` (value)
Setter for `switchVisibleAnimationDuration()`.
- `OnTabCreationRequested`
Triggered when the user requested a new tab.
- `addTab` (config)
Adds a new tab.
- `declareProperty` (k, defaultV)
Declares a widget property.
- `getProperty` (k)
General-purpose getter for widget properties.
- `setProperty` (k, v)
General-purpose setter for widget properties.
- `bindProperty` (k, vb)
Binds a `DataBinding` to a property. Read Manual for details about data bindings.
- `init` (rootNameScope)
Initializes the widget.
- `doinit` ()
Executes late widget initialization (i.e. after properties are applied).
- `doinitEarly` ()
Executes early widget initialization (i.e. before properties are applied).
- `resize` ()
Applies the new widget size to internal content.

- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (class)
Adds the css class `class` to this widget.
- [removeStyleClass](#) (class)
Removes the css class `class` from this widget.
- [isStyleClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children [clove::Widget](#) instances.
- [parentWidget](#) ()
The parent [clove::Widget](#).
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).

- **setEnabled** (value)
Setter for **enabled()**.
- **styleClass** ()
Custom css class(es).
- **setStyleClass** (value)
Setter for **styleClass()**.
- **style** ()
Custom css style string. You should not use that, but **styleClass()** instead.
- **setStyle** (value)
Setter for **style()**.
- **visibility** ()
If this widget is visible.
- **setVisibility** (value)
Setter for **visibility()**.
- **doStandaloneResizing** ()
If the widget handles to resize itself as needed.
- **setDoStandaloneResizing** (value)
Setter for **doStandaloneResizing()**.
- **mayFocus** ()
If the widget can have the keyboard focus.
- **setMayFocus** (value)
Setter for **mayFocus()**.
- **horizontalStretchAffinity** ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- **setHorizontalStretchAffinity** (value)
Setter for **horizontalStretchAffinity()**.
- **verticalStretchAffinity** ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- **setVerticalStretchAffinity** (value)
Setter for **verticalStretchAffinity()**.
- **strictHorizontalSizing** ()
If the widget is strictly forbidden to get additional horizontal space.
- **setStrictHorizontalSizing** (value)
Setter for **strictHorizontalSizing()**.
- **strictVerticalSizing** ()
If the widget is strictly forbidden to get additional vertical space.
- **setStrictVerticalSizing** (value)
Setter for **strictVerticalSizing()**.
- **vstretch** ()
Alias for **verticalStretchAffinity()**.
- **setVstretch** (value)
Setter for **vstretch()**.
- **hstretch** ()
Alias for **horizontalStretchAffinity()**.
- **setHstretch** (value)
Setter for **hstretch()**.
- **busy** ()
If the widget is in busy state, typically resulting in a loading animation.
- **setBusy** (value)
Setter for **busy()**.

- [registerBusy \(\)](#)
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy \(token\)](#)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus \(\)](#)
If the widget has the keyboard focus.
- [innerSize \(\)](#)
Returns inner width and height of this widget.
- [outerSize \(\)](#)
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.26.1 Detailed Description

A carousel container.

Something like a tab bar, but automatically switching forward tabs in regular intervals (and with a different style).

1.26.2 Member Function Documentation

1.26.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.26.2.2 addTab()

```
addTab (
    config ) [inherited]
```

Adds a new tab.

The configuration may contain the following additional keys:

- `tabLabel`: The tab header text.
- `tabIcon`: The tab icon as [clove::Icon](#).
- `tabMayBeClosedByUser`: If the user may close this tab.

This method returns a [clove::Widget](#) which can be used for getting subwidgets or removing the tab.

Parameters

<i>config</i>	A widget configuration like for clove::build() .
---------------	--

1.26.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.26.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.26.2.5 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.26.2.6 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.26.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.26.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.26.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.26.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.26.2.11 currentTab()

```
currentTab ( ) [inherited]
```

The index of the currently selected tab.

1.26.2.12 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.26.2.13 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.26.2.14 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.26.2.15 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.26.2.16 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.26.2.17 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.26.2.18 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.26.2.19 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.26.2.20 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.26.2.21 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.26.2.22 `getMinimalWidth()`

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.26.2.23 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.26.2.24 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.26.2.25 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.26.2.26 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.26.2.27 `horizontalStretchAffinity()`

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.26.2.28 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.26.2.29 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.26.2.30 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.26.2.31 interval()

```
interval ( )
```

The tab switching interval (in milliseconds).

1.26.2.32 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.26.2.33 isPlaying()

```
isPlaying ( )
```

If automatic tab switching is enabled.

1.26.2.34 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.26.2.35 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.26.2.36 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.26.2.37 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.26.2.38 next()

```
next ( )
```

Switches to the next tab.

1.26.2.39 OnDestroyed()

```
OnDestroyed [inherited]
```

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.26.2.40 OnKeyDown()

```
OnKeyDown [inherited]
```

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.26.2.41 OnKeyPress()

```
OnKeyPress [inherited]
```

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.26.2.42 OnKeyUp()

```
OnKeyUp [inherited]
```

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.26.2.43 OnPropertyChanged()

```
OnPropertyChanged [inherited]
```

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.26.2.44 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.26.2.45 OnTabCreationRequested()

OnTabCreationRequested [inherited]

Triggered when the user requested a new tab.

See also [userMayAddTabs\(\)](#).

This is a [clove.Event](#) instance. See Manual for details.

1.26.2.46 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.26.2.47 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.26.2.48 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.26.2.49 pause()

pause ()

Stops automatic tab switching.

1.26.2.50 play()

```
play ( )
```

Begins automatical tab switching.

1.26.2.51 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.26.2.52 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.26.2.53 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.26.2.54 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.26.2.55 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.26.2.56 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.57 `setCurrentTab()`

```
setCurrentTab (
    value ) [inherited]
```

Setter for [currentTab\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.58 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.59 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.60 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.61 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.62 setInterval()

```
setInterval (
    value )
```

Setter for [interval\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.63 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.64 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.65 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.26.2.66 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.67 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.68 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.69 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.70 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.26.2.71 `setSwitchInvisibleAnimationDuration()`

```
setSwitchInvisibleAnimationDuration (
    value ) [inherited]
```

Setter for [switchInvisibleAnimationDuration\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.72 `setSwitchInvisibleAnimationName()`

```
setSwitchInvisibleAnimationName (
    value ) [inherited]
```

Setter for [switchInvisibleAnimationName\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.73 setSwitchVisibleAnimationDuration()

```
setSwitchVisibleAnimationDuration (
    value ) [inherited]
```

Setter for [switchVisibleAnimationDuration\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.74 setSwitchVisibleAnimationName()

```
setSwitchVisibleAnimationName (
    value ) [inherited]
```

Setter for [switchVisibleAnimationName\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.75 setTabBarLocation()

```
setTabBarLocation (
    value ) [inherited]
```

Setter for [tabBarLocation\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.76 setTabs()

```
setTabs (
    value ) [inherited]
```

Setter for [tabs\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.77 setUserMayAddTabs()

```
setUserMayAddTabs (
    value ) [inherited]
```

Setter for [userMayAddTabs\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.78 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.79 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.80 `setVstretch()`

```
setVstretch (
    value ) [inherited]
```

Setter for `vstretch()`.

Parameters

<i>value</i>	The new value.
--------------	----------------

1.26.2.81 `strictHorizontalSizing()`

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.26.2.82 `strictVerticalSizing()`

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.26.2.83 `style()`

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.26.2.84 `styleClass()`

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.26.2.85 `switchInvisibleAnimationDuration()`

```
switchInvisibleAnimationDuration ( ) [inherited]
```

Optional animation duration (in msec) for the switch away animation.

See also [clove::TabView::switchInvisibleAnimationName\(\)](#).

1.26.2.86 `switchInvisibleAnimationName()`

```
switchInvisibleAnimationName ( ) [inherited]
```

Optional animation name for switching away from a tab.

1.26.2.87 `switchVisibleAnimationDuration()`

```
switchVisibleAnimationDuration ( ) [inherited]
```

Optional animation duration (in msec) for the switch animation.

See also [clove::TabView::switchVisibleAnimationName\(\)](#).

1.26.2.88 `switchVisibleAnimationName()`

```
switchVisibleAnimationName ( ) [inherited]
```

Optional animation name for switching to a tab.

1.26.2.89 `tabBarLocation()`

```
tabBarLocation ( ) [inherited]
```

Where to place the tab bar.

Either 'top', 'left', 'bottom' or 'right'.

1.26.2.90 `tabs()`

```
tabs ( ) [inherited]
```

The list of tabs.

Each tab is a widget configuration, like for [clove::build\(\)](#), with some additional optional keys which holds infos like the tab header text. So, for example, one item in that list could be `{view:'Something', ..., tabLabel:'Tab1'}`. See also [addTab\(\)](#).

1.26.2.91 `unregisterBusy()`

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.26.2.92 userMayAddTabs()

```
userMayAddTabs ( ) [inherited]
```

If to show a button for adding new tabs.

If `true`, implement a handler for `OnTabCreationRequested` and create a tab with [addTab\(\)](#) there.

1.26.2.93 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.26.2.94 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.26.2.95 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

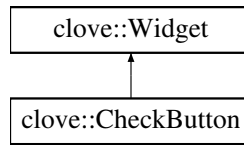
The documentation for this class was generated from the following file:

- `_meta/readme/clove.js`

1.27 clove::CheckBox Class Reference

A check button.

Inheritance diagram for clove::CheckBox:



Public Member Functions

- [label](#) ()
The label text.
- [setLabel](#) (value)
Setter for [label\(\)](#).
- [checked](#) ()
If this check button is checked.
- [setChecked](#) (value)
Setter for [checked\(\)](#).
- [OnChanged](#)
Triggered when this check button was selected.
- [OnClicked](#)
Triggered when this check button was selected. Same as OnChanged.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (class)
Adds the css class `class` to this widget.
- [removeStyleClass](#) (class)
Removes the css class `class` from this widget.
- [isStyleClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.

- [outerSize \(\)](#)
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.27.1 Detailed Description

A check button.

The user can choose to check and uncheck freely (in contrast to [clove::RadioButton](#)).

1.27.2 Member Function Documentation

1.27.2.1 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.27.2.2 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.27.2.3 `busy()`

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.27.2.4 `checked()`

```
checked ( )
```

If this check button is checked.

1.27.2.5 `childrenWidgets()`

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.27.2.6 `computeMinimalHeightForWidth()`

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.27.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.27.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.27.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.27.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.27.2.11 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.27.2.12 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.27.2.13 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.27.2.14 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.27.2.15 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.27.2.16 `effectivelyEnabled()`

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.27.2.17 `effectiveVisibility()`

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.27.2.18 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.27.2.19 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.27.2.20 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.27.2.21 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.27.2.22 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.27.2.23 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.27.2.24 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.27.2.25 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.27.2.26 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.27.2.27 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.27.2.28 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.27.2.29 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.27.2.30 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.27.2.31 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.27.2.32 label()

```
label ( )
```

The label text.

1.27.2.33 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.27.2.34 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.27.2.35 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.27.2.36 OnChanged()

`OnChanged`

Triggered when this check button was selected.

This is a [clove.Event](#) instance. See Manual for details.

1.27.2.37 OnClicked()

`OnClicked`

Triggered when this check button was selected. Same as OnChanged.

This is a [clove.Event](#) instance. See Manual for details.

1.27.2.38 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.27.2.39 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.27.2.40 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.27.2.41 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.27.2.42 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.27.2.43 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.27.2.44 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.27.2.45 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.27.2.46 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.27.2.47 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.27.2.48 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.27.2.49 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.27.2.50 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.27.2.51 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.27.2.52 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.53 `setChecked()`

```
setChecked (
    value )
```

Setter for [checked\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.54 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.55 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.56 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.57 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.58 `setLabel()`

```
setLabel (
    value )
```

Setter for [label\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.59 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.60 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.61 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.27.2.62 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (  
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.63 setStrictVerticalSizing()

```
setStrictVerticalSizing (  
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.64 setStyle()

```
setStyle (  
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.65 setStyleClass()

```
setStyleClass (  
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.66 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.27.2.67 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.68 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.69 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.27.2.70 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.27.2.71 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.27.2.72 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.27.2.73 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.27.2.74 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.27.2.75 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.27.2.76 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.27.2.77 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.28 clove Class Reference

Clove root namespace.

Classes

- class [AbstractMenu](#)
A base class for widgets which present a menu of actions to the user.
- class [AjaxAsyncDatasource](#)
A [clove::AsyncDatasource](#) implementation which uses Ajax requests for data retrieval and modification.
- class [AsyncDatasource](#)
A [clove::Datasource](#) implementation which supports asynchronous data operations like Ajax requests.
- class [AudioPlayer](#)
A widget which plays an audio source and optionally allows user controls.
- class [Border](#)
A single-widget container framing the inner one with (a css-styled) border.
- class [Button](#)
A button allows the user to trigger a particular program event.
- class [Carousel](#)
A carousel container.
- class [CheckButton](#)
A check button.
- class [DataBinding](#)
A data binding.
- class [DataBindingConverter](#)
A data binding converter maps values between ui representation and data model in a custom way.
- class [Datasource](#)
Base class for datasources.
- class [DatasourceValuePointer](#)
A pointer to one node in a [clove::Datasource](#).
- class [DataView](#)
Base class for some views which shows hierarchical data from a [clove::Datasource](#).
- class [DateBox](#)
A user input box for date input.
- class [Dialog](#)
A dialog is a container for a new sub user interface, decorated with a border and a title bar.
- class [DropDownBox](#)
A single-line text box (without edit functionality) with a popup list of values to select from.
- class [EditBox](#)
A single-line box for text input from the user.
- class [EditComboBox](#)
A single-line box for text input from the user with an additional popup list of value recommendations.
- class [Event](#)
An event. It can have some handlers (or: listeners) and can be triggered.
- class [EventArgs](#)
Arguments for a particular incidence of a [clove::Event](#).
- class [Expander](#)
A single-widget container which can be expanded or collapsed.
- class [FilterProxyDatasource](#)
A [clove::Datasource](#) implementation which filters another [clove::Datasource](#).
- class [FlatLayout](#)
A mixin for flat layout implementations.
- class [FlatView](#)
A container which shows one of its child widgets in full size and hides all others.
- class [Form](#)

- A container for a form-like with a label for each entry in a row-oriented alignment.*

 - class [Grid](#)
- A container for aligning child widgets in a grid.*

 - class [GridLayout](#)
- A mixin for grid layout implementations.*

 - class [Headersource](#)
- Base class for headersources.*

 - class [HorizontalStack](#)
- A container which stacks child widgets column-wise.*

 - class [HtmlView](#)
- A host for arbitrary html content.*

 - class [I18N](#)
- Internationalization support class.*

 - class [Icon](#)
- An icon.*

 - class [IconView](#)
- Shows an icon.*

 - class [ImageView](#)
- Shows an image.*

 - class [Label](#)
- A text label.*

 - class [Layout](#)
- A mixin for a widget classes which align child widgets in some way.*

 - class [ListView](#)
- A view which shows a [clove::Datasource](#) in a list.*

 - class [MainView](#)
- A typical main view, including a [clove::Toolbar](#) and a container for an inner body widget.*

 - class [MediaPlayer](#)
- A base class for media player widgets.*

 - class [Menubar](#)
- A menu bar.*

 - class [ModalPanel](#)
- A surface for implementing modality.*

 - class [MultilineEditBox](#)
- A multi-line box for text input from the user.*

 - class [NameScope](#)
- A mixin realizing a new namespace.*

 - class [NativeDatasource](#)
- A [clove::Datasource](#) implementation which stores all data in-memory.*

 - class [NativeDatasourceNode](#)
- A node value implementation for [clove::NativeDatasource](#).*

 - class [NotificationController](#)
- Controller for clove notifications.*

 - class [NumericEditBox](#)
- A text box with up/down buttons for numbers.*

 - class [PasswordEditBox](#)
- A box for password input from the user.*

 - class [PopupMenu](#)
- A popup menu (i.e. a box with vertically listed actions).*

 - class [PopupMenuButton](#)
- A special [clove::Button](#) which opens a popup menu when it is triggered.*

- class [ProgressBar](#)
A progress bar.
- class [ProxyDatasource](#)
A [Clove::Datasource](#) implementation which proxies the content of another datasource in a somehow transformed way.
- class [RadioButton](#)
A radio button.
- class [RadioGroup](#)
Groups some [Clove::RadioButton](#) together in a logical way, so the user can select exactly one of them.
- class [RawResizeSplitter](#)
The actual splitter in a [Clove::ResizeSplitter](#).
- class [ResizeSplitter](#)
A container for multiple widgets with splitters for manual resizing between them.
- class [RichEdit](#)
A user input box for text with rich formatting.
- class [RichEditBox](#)
A user input box for text with rich formatting.
- class [RootNameScope](#)
A standalone namespace.
- class [ScrollView](#)
A container for making the children scrollable.
- class [Slider](#)
A slider allows the user to choose a number from a given value interval.
- class [SortProxyDatasource](#)
A [Clove::Datasource](#) implementation which sorts another [Clove::Datasource](#).
- class [Spacer](#)
A spacer widget for filling gaps in a layout.
- class [StackLayout](#)
A mixin for stack layout implementations.
- class [symbols](#)
Namespace for symbol constants.
- class [TableView](#)
A view which shows a [Clove::Datasource](#) in a table.
- class [TabView](#)
A tabbed container.
- class [TimeBox](#)
A user input box for time input.
- class [Toolbar](#)
A toolbar has header texts and a menu bar in a perceptible visual bar, typically used on top of a user interface with all available width.
- class [TreeView](#)
A view which shows a [Clove::Datasource](#) in a tree.
- class [utils](#)
Namespace for some utilities.
- class [VerticalStack](#)
A container which stacks child widgets row-wise.
- class [VideoPlayer](#)
A widget which plays a video source and optionally allows user controls.
- class [Widget](#)
Base class for a Clove widget.
- class [Wrap](#)
A container for aligning child widgets in horizontal wrapping lines.
- class [WrapLayout](#)
A mixin for wrap layout implementations.

Public Member Functions

- [build](#) (config, buildconfig)
Constructs new widgets according to a widget configuration and controlled by the build configuration.
- [getByName](#) (name)
Finds a widget by name in the clove root namespace.
- [populateUI](#) (initfct, populateconfig)
Utility for user interface creation.
- [conversationDialog](#) (config)
Shows a conversation dialog (small and easy dialog box).
- [messageDialog](#) (config)
Shows a message dialog.
- [inputDialog](#) (config)
Shows an input dialog.
- [selectionDialog](#) (config)
Shows a dialog with a list of choices.
- [Visible](#)
Visibility value for a visible widget.
- [Invisible](#)
Visibility value for a widget, which is invisible but occupies its room.
- [InvisibleCollapsed](#)
Visibility value for a widget, which is not visible and does not take any space.
- [i18n](#)
Provides internationalization features.
- [rootNameScope](#)
The Clove root namespace.
- [browser](#)
Some data helpful for browser detection.
- [databind](#) (config)
Constructs a [clove.DataBinding](#).
- [MenuSeparator](#)
A menu separator for usage in [clove::AbstractMenu::actions\(\)](#).
- [notifications](#)
The clove notification controller.

1.28.1 Detailed Description

Clove root namespace.

1.28.2 Member Function Documentation

1.28.2.1 [browser\(\)](#)

`browser`

Some data helpful for browser detection.

1.28.2.2 build()

```
build (
    config,
    buildconfig )
```

Constructs new widgets according to a widget configuration and controlled by the build configuration.

Read the Manual for details.

Parameters

<i>config</i>	The widget configuration; like a blueprint for the new widget.
<i>buildconfig</i>	The build configuration; specifies some aspects about the construction.

1.28.2.3 conversationDialog()

```
conversationDialog (
    config )
```

Shows a conversation dialog (small and easy dialog box).

This builds a small [clove::Dialog](#) with an arbitrary body widget and a button bar. It returns a dialog result structure, as [clove::utils::popup\(\)](#), also containing `OnProceed`. This [clove::Event](#) triggers when the user clicks on one of the buttons in the button bar.

config may contain:

- `title`: [Dialog](#) title text.
- `body`: Inner widget as widget configuration like for [clove::build\(\)](#).
- `icon`: [Icon](#) as [clove::Icon](#).
- `buttons`: List of button label strings.

Parameters

<i>config</i>	A conversation dialog configuration object.
---------------	---

1.28.2.4 databind()

```
databind (
    config )
```

Constructs a [clove.DataBinding](#).

The returned binding is for usage in a widget configuration. It is not completely configured, so it will not work out of the box in other situations.

The databind configuration may contain the following parameters:

- `datasource`: The [clove.Datasource](#) instance to bind.
- `row`: The row index.
- `col`: The column index.
- `parent`: The parent node as [clove::DatasourceValuePointer](#).
- `dataDirection`: One of 'todatasource', 'towidget' or 'bidirectional'.
- `valueComparator`: A `function(a,b)` which decides if two values are considered as equal. Default is something like `a==b`, but also looks for an `equals` method on the objects.

Parameters

<i>config</i>	The databind configuration.
---------------	-----------------------------

- `valueConverter`: A [clove::DataBindingConverter](#) which translates between datasource value and user interface representation. Default is a noop conversion.

1.28.2.5 `getByName()`

```
getByName (
    name )
```

Finds a widget by name in the clove root namespace.

Read the Manual for details.

Parameters

<i>name</i>	The widget name.
-------------	------------------

1.28.2.6 `i18n()`

```
i18n
```

Provides internationalization features.

Read the Manual for details.

1.28.2.7 inputDialog()

```
inputDialog (
    config )
```

Shows an input dialog.

This builds a small `clove::Dialog` which asks the user for some text input. It returns an `clove::Event` which is triggered when the user finished.

config may contain:

- `title`: `Dialog` title text.
- `question`: The question text.
- `buttons`: List of button label strings.
- `defaultAnswer`: The initial answer string.

Note: In very exotic situations, it might be useful to access the complete dialog result structure as `clove↔::conversationDialog()` would return. This is possible with the `dlgres` property of the returned object.

Parameters

<code>config</code>	A conversation dialog configuration object.
---------------------	---

1.28.2.8 Invisible()

```
Invisible
```

Visibility value for a widget, which is invisible but occupies its room.

1.28.2.9 InvisibleCollapsed()

```
InvisibleCollapsed
```

Visibility value for a widget, which is not visible and does not take any space.

1.28.2.10 MenuSeparator()

```
MenuSeparator
```

A menu separator for usage in `clove::AbstractMenu::actions()`.

1.28.2.11 `messageDialog()`

```
messageDialog (
    config )
```

Shows a message dialog.

This builds a small `clove::Dialog` which contains a message and asks the user to click on a button. It returns an `clove::Event` which is triggered when the user clicked on a button.

`config` may contain:

- `title`: `Dialog` title text.
- `message`: The message text.
- `buttons`: List of button label strings.

Note: In very exotic situations, it might be useful to access the complete dialog result structure as `clove↔::conversationDialog()` would return. This is possible with the `dlgres` property of the returned object.

Parameters

<i>config</i>	A conversation dialog configuration object.
---------------	---

1.28.2.12 `notifications()`

```
notifications
```

The clove notification controller.

It is an instance of `clove::NotificationController`.

1.28.2.13 `populateUI()`

```
populateUI (
    initfct,
    populateconfig )
```

Utility for user interface creation.

Using this function for building the main user interface (or other complex pieces) enhances the performance and can provide a loading indicator visible to the user.

Parameters

<i>initfct</i>	A function which actually builds the user interface.
<i>populateconfig</i>	A (optional) configuration object which specifies some behavior aspects during initialization.

1.28.2.14 rootNameScope()

`rootNameScope`

The Clove root namespace.

1.28.2.15 selectionDialog()

```
selectionDialog (
    config )
```

Shows a dialog with a list of choices.

This builds a small [clove::Dialog](#) which asks the user to choose an item from a list. It returns an [clove::Event](#) which is triggered when the user finished.

`config` may contain:

- `title`: [Dialog](#) title text.
- `question`: The question text.
- `buttons`: List of button label strings.
- `defaultAnswer`: The initial answer string.
- `choices`: The list of choice strings.

Note: In very exotic situations, it might be useful to access the complete dialog result structure as [clove↔::conversationDialog\(\)](#) would return. This is possible with the `dlgres` property of the returned object.

Parameters

<code>config</code>	A conversation dialog configuration object.
---------------------	---

1.28.2.16 Visible()

`Visible`

Visibility value for a visible widget.

The documentation for this class was generated from the following file:

- `_meta/readme/clove.js`

1.29 clove::DataBinding Class Reference

A data binding.

Public Member Functions

- [DataBinding](#) (widget, propertyName, datasource, ptr, dataDirection, valueComparator, valueConverter)
- [bind](#) ()
Activates the binding.
- [unbind](#) ()
Deactivates the binding.

1.29.1 Detailed Description

A data binding.

Read the Manual for details.

1.29.2 Constructor & Destructor Documentation

1.29.2.1 DataBinding()

```
DataBinding (
    widget,
    propertyName,
    datasource,
    ptr,
    dataDirection,
    valueComparator,
    valueConverter )
```

Parameters

<i>widget</i>	The clove::Widget to bind to the datasource.
<i>propertyName</i>	The property name in the widget to bind.
<i>datasource</i>	The clove::Datasource to bind to the widget.
<i>ptr</i>	The clove::DatasourceValuePointer .
<i>dataDirection</i>	See clove::databind() .
<i>valueComparator</i>	Optional custom function which checks if two values are equal.
<i>valueConverter</i>	Optional clove::DataBindingConverter for conversion between ui and data model.

1.29.3 Member Function Documentation

1.29.3.1 bind()

```
bind ( )
```

Activates the binding.

1.29.3.2 unbind()

```
unbind ( )
```

Deactivates the binding.

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.30 clove::DataBindingConverter Class Reference

A data binding converter maps values between ui representation and data model in a custom way.

Public Member Functions

- [convertTo](#) (x)
Converts an original datasource value to a user interface representation.
- [convertFrom](#) (y)
Converts a user interface representation to an original datasource value.

1.30.1 Detailed Description

A data binding converter maps values between ui representation and data model in a custom way.

1.30.2 Member Function Documentation

1.30.2.1 convertFrom()

```
convertFrom (  
    y )
```

Converts a user interface representation to an original datasource value.

Parameters

<code>y</code>	The user interface representation value.
----------------	--

1.30.2.2 convertTo()

```
convertTo (
    x )
```

Converts an original datasource value to a user interface representation.

Parameters

<code>x</code>	The original datasource value.
----------------	--------------------------------

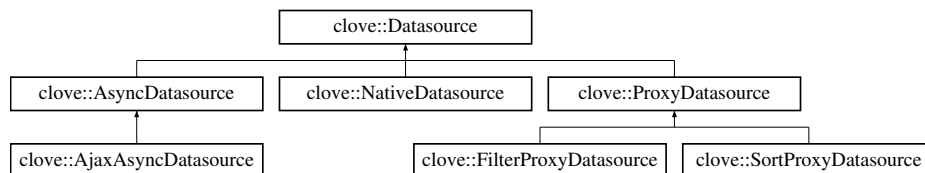
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.31 clove::Datasource Class Reference

Base class for datasources.

Inheritance diagram for clove::Datasource:



Public Member Functions

- [getValue](#) (`ptr`)
Returns the value for a given node.
- [getMetadata](#) (`ptr`)
Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.
- [changeValue](#) (`ptr`, `value`)
Change the value for a given node.
- [rowCount](#) (`parent`)
Returns the number of rows for a given node.
- [columnCount](#) (`parent`)
Returns the number of columns for a given node.
- [valuePointer](#) (`irow`, `icol`, `parent`)

Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.

- [parent](#) (ptr)

Returns the parent node for a given node.

- [valuePointerNavigateInDepth](#) (ptr, direction, mayexpandfct)

Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.

- [OnDataInsert](#)

Triggered when a node insertion takes place.

- [OnDataRemove](#)

Triggered when a node removal takes place.

- [OnDataUpdate](#)

Triggered when a node data update takes place.

1.31.1 Detailed Description

Base class for datasources.

Read the Manual for details.

1.31.2 Member Function Documentation

1.31.2.1 `changeValue()`

```
changeValue (
    ptr,
    value ) [pure virtual]
```

Change the value for a given node.

This method is called from external places, e.g. when a user makes changes in a datasource-connected widget.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>value</i>	The new value.

1.31.2.2 `columnCount()`

```
columnCount (
    parent ) [pure virtual]
```

Returns the number of columns for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.31.2.3 getMetadata()

```
getMetadata (
    ptr ) [pure virtual]
```

Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.31.2.4 getValue()

```
getValue (
    ptr ) [pure virtual]
```

Returns the value for a given node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.31.2.5 OnDataInsert()

```
OnDataInsert
```

Triggered when a node insertion takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.31.2.6 OnDataRemove()

```
OnDataRemove
```

Triggered when a node removal takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.31.2.7 OnDataUpdate()

OnDataUpdate

Triggered when a node data update takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.31.2.8 parent()

```
parent (
    ptr ) [pure virtual]
```

Returns the parent node for a given node.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
------------	---

1.31.2.9 rowCount()

```
rowCount (
    parent ) [pure virtual]
```

Returns the number of rows for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.31.2.10 valuePointer()

```
valuePointer (
    irow,
    icol,
    parent ) [pure virtual]
```

Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.

Parameters

<i>irow</i>	The row index.
<i>icol</i>	The column index.
<i>parent</i>	The parent as clove::DatasourceValuePointer .

1.31.2.11 valuePointerNavigateInDepth()

```
valuePointerNavigateInDepth (
    ptr,
    direction,
    mayexpandfct )
```

Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
<i>direction</i>	The direction (+1 or -1).
<i>mayexpandfct</i>	A function(<i>ptr</i>) which returns <code>true</code> iff this node's children are to be traversed.

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.32 clove::DatasourceValuePointer Class Reference

A pointer to one node in a [clove::Datasource](#).

Public Member Functions

- [DatasourceValuePointer](#) (*irow*, *icol*, *backendObject*, *datasource*)
- [irow](#)
The row index.
- [icol](#)
The column index.
- [datasource](#)
The owning [clove::Datasource](#).
- [backendObject](#)
The backend object.
- [parent](#) ()
Returns the parent [clove::DatasourceValuePointer](#).
- [sibling](#) (*irow*, *icol*)
Returns a sibling [clove::DatasourceValuePointer](#).
- [rowCount](#) ()
Returns the number of rows in this node.
- [columnCount](#) ()
Returns the number of columns in this node.
- [getValue](#) ()
Returns the value stored behind this pointer.
- static [equals](#) (*ptr1*, *ptr2*)

- *Checks if two value pointers point to the same place.*
- static `toPath` (`ptr`)
Returns an array-of-tuples representation for a value pointer.
- static `fromPath` (`datasource`, `path`)
Returns a `clove::DatasourceValuePointer` for a array-of-tuples representation.
- static `toString` (`ptr`)
Returns a string representation for a value pointer.

1.32.1 Detailed Description

A pointer to one node in a `clove::Datasource`.

1.32.2 Constructor & Destructor Documentation

1.32.2.1 DatasourceValuePointer()

```
DatasourceValuePointer (
    irow,
    icol,
    backendObject,
    datasource )
```

Typically called only inside a `clove::Datasource` implementation. From outside, you should use `clove::Datasource←
::valuePointer`.

Parameters

<code>irow</code>	The row index.
<code>icol</code>	The column index.
<code>backendObject</code>	The backend object. This depends on the datasource implementation.
<code>datasource</code>	The owning <code>clove::Datasource</code> .

1.32.3 Member Function Documentation

1.32.3.1 backendObject()

`backendObject`

The backend object.

This depends on the datasource implementation.

1.32.3.2 columnCount()

```
columnCount ( )
```

Returns the number of columns in this node.

1.32.3.3 datasource()

```
datasource
```

The owning [clove::Datasource](#).

1.32.3.4 equals()

```
static equals (
    ptr1,
    ptr2 )
```

Checks if two value pointers point to the same place.

Parameters

<i>ptr1</i>	The first clove::DatasourceValuePointer .
<i>ptr2</i>	The second clove::DatasourceValuePointer .

1.32.3.5 fromPath()

```
static fromPath (
    datasource,
    path )
```

Returns a [clove::DatasourceValuePointer](#) for a array-of-tuples representation.

See also [clove::DatasourceValuePointer::toPath\(\)](#).

Parameters

<i>datasource</i>	The clove::Datasource the value pointer shall correspond to.
<i>path</i>	The array-of-tuples.

1.32.3.6 `getValue()`

```
getValue ( )
```

Returns the value stored behind this pointer.

1.32.3.7 `icol()`

```
icol
```

The column index.

1.32.3.8 `irow()`

```
irow
```

The row index.

1.32.3.9 `parent()`

```
parent ( )
```

Returns the parent [clove::DatasourceValuePointer](#).

1.32.3.10 `rowCount()`

```
rowCount ( )
```

Returns the number of rows in this node.

1.32.3.11 `sibling()`

```
sibling (
    irow,
    icol )
```

Returns a sibling [clove::DatasourceValuePointer](#).

Parameters

<i>irow</i>	The row index.
<i>icol</i>	The column index.

1.32.3.12 toPath()

```
static toPath (  
    ptr )
```

Returns an array-of-tuples representation for a value pointer.

See also [clove::DatasourceValuePointer::fromPath\(\)](#).

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer to convert.
------------	---

1.32.3.13 toString()

```
static toString (  
    ptr )
```

Returns a string representation for a value pointer.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

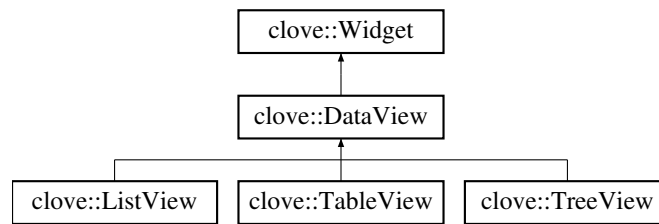
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.33 clove::DataView Class Reference

Base class for some views which shows hierarchical data from a [clove::Datasource](#).

Inheritance diagram for [clove::DataView](#):



Public Member Functions

- [datasource](#) ()
The [clove::Datasource](#) for this view.
- [setDatasource](#) (value)
Setter for [datasource\(\)](#).
- [dataViewProcessor](#) ()
An optional function (ptr, value) for processing a datasource value to a display string.
- [setDataViewProcessor](#) (value)
Setter for [dataViewProcessor\(\)](#).
- [cellGenerator](#) ()
A generator function for complex cell content.
- [setCellGenerator](#) (value)
Setter for [cellGenerator\(\)](#).
- [alwaysAllocateExpanderSpace](#) ()
If some space is allocated for an expander even for cells without children.
- [setAlwaysAllocateExpanderSpace](#) (value)
Setter for [alwaysAllocateExpanderSpace\(\)](#).
- [showOnlyFirstColumn](#) ()
If to show only the first column and hide the other ones.
- [setShowOnlyFirstColumn](#) (value)
Setter for [showOnlyFirstColumn\(\)](#).
- [hideExpanders](#) ()
If to hide all expanders.
- [setHideExpanders](#) (value)
Setter for [hideExpanders\(\)](#).
- [allowSelection](#) ()
If it is allowed to select nodes.
- [setAllowSelection](#) (value)
Setter for [allowSelection\(\)](#).
- [allowChecking](#) ()
If it is allowed to check cells.
- [setAllowChecking](#) (value)
Setter for [allowChecking\(\)](#).
- [granularity](#) ()
The granularity for stuff like selection and checking.
- [setGranularity](#) (value)
Setter for [granularity\(\)](#).
- [showChangeMenu](#) ()
If a menu shall be shown for making manual changes to the data source.
- [setShowChangeMenu](#) (value)
Setter for [showChangeMenu\(\)](#).

- [editOnGesture](#) ()
If the user shall be able to edit cells by double clicking/tapping them.
- [setEditOnGesture](#) (value)
Setter for [editOnGesture\(\)](#).
- [rowsResizable](#) ()
If the user shall be able to resize rows.
- [setRowsResizable](#) (value)
Setter for [rowsResizable\(\)](#).
- [columnsResizable](#) ()
If the user shall be able to resize columns.
- [setColumnsResizable](#) (value)
Setter for [columnsResizable\(\)](#).
- [selectCell](#) (ptr)
Selects a cell.
- [isCellSelected](#) (ptr)
Checks if a given cell is selected.
- [checkedCells](#) ()
Returns the checked cells as list of [clove::DatasourceValuePointer](#).
- [setCheckedCells](#) (ptrs)
Sets the checked cells.
- [isCellChecked](#) (ptr)
Checks if a given cell is checked.
- [setCellChecked](#) (ptr, val)
Sets if a given cell is checked.
- [selection](#) ()
Returns the selected item as [clove::DatasourceValuePointer](#).
- [headersource](#) ()
The [clove::Headersource](#) providing row and column header configurations.
- [setHeadersource](#) (value)
Setter for [headersource\(\)](#).
- [gridVisible](#) ()
If to show a cell grid.
- [setGridVisible](#) (value)
Setter for [gridVisible\(\)](#).
- [nodeActivationNeedsDoubleClick](#) ()
If node activation needs a double-click.
- [setNodeActivationNeedsDoubleClick](#) (value)
Setter for [nodeActivationNeedsDoubleClick\(\)](#).
- [expandCell](#) (ptr)
Expands a given cell.
- [expandCellRecursive](#) (ptr)
Expands a given cell and all parents.
- [collapseCell](#) (ptr)
Collapses a given cell.
- [isCellExpanded](#) (ptr)
Checks if a given cell is expanded.
- [editCell](#) (ptr)
Triggers the edit mode for a cell.
- [OnSelectionChanged](#)
Triggered when the selection in the view changes.
- [declareProperty](#) (k, defaultV)

- Declares a widget property.*

 - `getProperty` (k)
General-purpose getter for widget properties.
 - `setProperty` (k, v)
General-purpose setter for widget properties.
 - `bindProperty` (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
 - `init` (rootNameScope)
Initializes the widget.
 - `doinit` ()
Executes late widget initialization (i.e. after properties are applied).
 - `doinitEarly` ()
Executes early widget initialization (i.e. before properties are applied).
 - `resize` ()
Applies the new widget size to internal content.
 - `doresize` ()
Corrects alignments of internal elements according to the new widget size.
 - `relayout` ()
Notifies the parent widget that a new geometry is required.
 - `focus` ()
Sets the focus to this widget.
 - `isAlive` ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
 - `computeMinimalWidth` ()
Computes the minimal width in pixel this widget needs to have.
 - `computePreferredWidth` ()
Computes the preferred width in pixel this widget needs to have.
 - `computeMinimalHeightForWidth` (w)
Computes the minimal height in pixel this widget needs to have for a given width.
 - `computePreferredHeightForWidth` (w)
Computes the preferred height in pixel this widget needs to have for a given width.
 - `getMinimalWidth` ()
Returns the minimal width in pixel this widget needs to have.
 - `getPreferredWidth` ()
Returns the preferred width in pixel this widget needs to have.
 - `getMinimalHeightForWidth` (w)
Returns the minimal height in pixel this widget needs to have for a given width.
 - `getPreferredHeightForWidth` (w)
Returns the preferred height in pixel this widget needs to have for a given width.
 - `addStyleClass` (css)
Adds the css class `css` to this widget.
 - `removeStyleClass` (css)
Removes the css class `css` from this widget.
 - `isStyleClass` (css)
Checks if this widget contains the css class `css`.
 - `setStyleClassAssigned` (css, assigned)
Sets or unsets the css class `css` to this widget.
 - `containingNameScope` ()
The namespace this widget contains to.
 - `nameScope` ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).

- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children [clove::Widget](#) instances.
- [parentWidget](#) ()
The parent [clove::Widget](#).
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()

- If the widget is strictly forbidden to get additional vertical space.*

 - **setStrictVerticalSizing** (value)
Setter for [strictVerticalSizing\(\)](#).
 - **vstretch** ()
Alias for [verticalStretchAffinity\(\)](#).
 - **setVstretch** (value)
Setter for [vstretch\(\)](#).
 - **hstretch** ()
Alias for [horizontalStretchAffinity\(\)](#).
 - **setHstretch** (value)
Setter for [hstretch\(\)](#).
 - **busy** ()
If the widget is in busy state, typically resulting in a loading animation.
 - **setBusy** (value)
Setter for [busy\(\)](#).
 - **registerBusy** ()
Sets the widget to busy state and returns a token which helps returning to normal state.
 - **unregisterBusy** (token)
Drops a token and returns to normal state if no other tokens exist.
 - **hasFocus** ()
If the widget has the keyboard focus.
 - **innerSize** ()
Returns inner width and height of this widget.
 - **outerSize** ()
Returns outer width and height of this widget.
 - **OnDestroyed**
Triggered when this widget was removed somehow.
 - **OnResized**
Triggered when this widget was resized.
 - **OnVisibilityChanged**
Triggered when the visibility of this widget changed.
 - **OnPropertyChanged**
Triggered whenever a property of this widget changed its value.
 - **OnKeyDown**
Triggered when a keyboard key went down.
 - **OnKeyUp**
Triggered when a keyboard key came up.
 - **OnKeyPress**
Triggered when a keyboard key was pressed.

1.33.1 Detailed Description

Base class for some views which shows hierarchical data from a [clove::Datasource](#).

Read the Manual for details.

Although this is a fully-functional and implemented class, you should consider using one of the subclasses.

1.33.2 Member Function Documentation

1.33.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.33.2.2 allowChecking()

```
allowChecking ( )
```

If it is allowed to check cells.

This is a way to select 0..n data cells. For a single-selection, please check [allowSelection\(\)](#).

1.33.2.3 allowSelection()

```
allowSelection ( )
```

If it is allowed to select nodes.

This is always a single selection. For something like multi-selection, please check [allowChecking\(\)](#).

1.33.2.4 alwaysAllocateExpanderSpace()

```
alwaysAllocateExpanderSpace ( )
```

If some space is allocated for an expander even for cells without children.

1.33.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.33.2.6 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.33.2.7 cellGenerator()

```
cellGenerator ( )
```

A generator function for complex cell content.

This method is called for each cell with ([clove::Widget](#), [clove::DatasourceValuePointer](#)). It may return a widget configuration as for [clove::build\(\)](#). If it does, the cell is constructed with this widget as content. The content must define an `update(clove::Widget, clove::DatasourceValuePointer, value)` method, which takes the cell datasource value and configures the inner widget according to that.

1.33.2.8 checkedCells()

```
checkedCells ( )
```

Returns the checked cells as list of [clove::DatasourceValuePointer](#).

1.33.2.9 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.33.2.10 collapseCell()

```
collapseCell (
    ptr )
```

Collapses a given cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.33.2.11 columnsResizable()

```
columnsResizable ( )
```

If the user shall be able to resize columns.

1.33.2.12 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.33.2.13 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.33.2.14 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.33.2.15 `computePreferredWidth()`

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.33.2.16 `containingNameScope()`

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.33.2.17 `datasource()`

```
datasource ( )
```

The [clove::Datasource](#) for this view.

This provides the actual data to display. See [clove::NativeDatasource](#) for a ready-to-use implementation.

1.33.2.18 `dataViewProcessor()`

```
dataViewProcessor ( )
```

An optional `function(ptr, value)` for processing a datasource value to a display string.

1.33.2.19 `declareProperty()`

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.33.2.20 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.33.2.21 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.33.2.22 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.33.2.23 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.33.2.24 editCell()

```
editCell (
    ptr )
```

Triggers the edit mode for a cell.

Only works if a method `_getdomcell(ir, ic, parent)` returns a dom node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.33.2.25 editOnGesture()

```
editOnGesture ( )
```

If the user shall be able to edit cells by double clicking/tapping them.

1.33.2.26 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.33.2.27 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.33.2.28 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.33.2.29 expandCell()

```
expandCell (
    ptr )
```

Expands a given cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.33.2.30 expandCellRecursive()

```
expandCellRecursive (
    ptr )
```

Expands a given cell and all parents.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.33.2.31 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.33.2.32 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.33.2.33 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.33.2.34 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.33.2.35 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.33.2.36 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.33.2.37 `granularity()`

```
granularity ( )
```

The granularity for stuff like selection and checking.

Either `'cell'` or `'row'`.

1.33.2.38 gridVisible()

`gridVisible ()`

If to show a cell grid.

1.33.2.39 hasFocus()

`hasFocus ()` [inherited]

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.33.2.40 headersource()

`headersource ()`

The [clove::Headersource](#) providing row and column header configurations.

1.33.2.41 hideExpanders()

`hideExpanders ()`

If to hide all expanders.

1.33.2.42 horizontalStretchAffinity()

`horizontalStretchAffinity ()` [inherited]

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.33.2.43 hstretch()

`hstretch ()` [inherited]

Alias for [horizontalStretchAffinity\(\)](#).

1.33.2.44 init()

`init (
 rootNameScope)` [inherited]

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.33.2.45 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.33.2.46 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.33.2.47 isCellChecked()

```
isCellChecked (
    ptr )
```

Checks if a given cell is checked.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.33.2.48 isCellExpanded()

```
isCellExpanded (
    ptr )
```

Checks if a given cell is expanded.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.33.2.49 isCellSelected()

```
isCellSelected (
    ptr )
```

Checks if a given cell is selected.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.33.2.50 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.33.2.51 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.33.2.52 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.33.2.53 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.33.2.54 nodeActivationNeedsDoubleClick()

`nodeActivationNeedsDoubleClick ()`

If node activation needs a double-click.

Note: If you set this, you should find alternative ways for users of mobile devices!

1.33.2.55 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.33.2.56 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.33.2.57 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.33.2.58 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.33.2.59 OnPropertyChanged()

`OnPropertyChanged [inherited]`

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.33.2.60 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.33.2.61 OnSelectionChanged()

OnSelectionChanged

Triggered when the selection in the view changes.

This is a [clove.Event](#) instance. See Manual for details.

1.33.2.62 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.33.2.63 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.33.2.64 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.33.2.65 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.33.2.66 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.33.2.67 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.33.2.68 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.33.2.69 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.33.2.70 rowsResizable()

```
rowsResizable ( )
```

If the user shall be able to resize rows.

1.33.2.71 selectCell()

```
selectCell (
    ptr )
```

Selects a cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.33.2.72 selection()

```
selection ( ) [pure virtual]
```

Returns the selected item as [clove::DatasourceValuePointer](#).

1.33.2.73 setAllowChecking()

```
setAllowChecking (
    value )
```

Setter for [allowChecking\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.74 setAllowSelection()

```
setAllowSelection (
    value )
```

Setter for [allowSelection\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.75 setAlwaysAllocateExpanderSpace()

```
setAlwaysAllocateExpanderSpace (
    value )
```

Setter for [alwaysAllocateExpanderSpace\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.76 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.77 setCellChecked()

```
setCellChecked (
    ptr,
    val )
```

Sets if a given cell is checked.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>val</i>	If the cells shall be checked.

1.33.2.78 setCellGenerator()

```
setCellGenerator (
    value )
```

Setter for [cellGenerator\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.79 setCheckedCells()

```
setCheckedCells (
    ptrs )
```

Seturns the checked cells.

Parameters

<i>ptrs</i>	A list of clove::DatasourceValuePointer .
-------------	---

1.33.2.80 setColumnsResizable()

```
setColumnsResizable (
    value )
```

Setter for [columnsResizable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.81 setDatasource()

```
setDatasource (
    value )
```

Setter for [datasource\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.82 setDataViewProcessor()

```
setDataViewProcessor (  
    value )
```

Setter for [dataViewProcessor\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.83 setDoStandaloneResizing()

```
setDoStandaloneResizing (  
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.84 setEditOnGesture()

```
setEditOnGesture (  
    value )
```

Setter for [editOnGesture\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.85 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.86 setGranularity()

```
setGranularity (
    value )
```

Setter for [granularity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.87 setGridVisible()

```
setGridVisible (
    value )
```

Setter for [gridVisible\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.88 setHeadersource()

```
setHeadersource (
    value )
```

Setter for [headersource\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.89 setHideExpanders()

```
setHideExpanders (
    value )
```

Setter for [hideExpanders\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.90 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.91 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.92 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.93 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.94 setNodeActivationNeedsDoubleClick()

```
setNodeActivationNeedsDoubleClick (
    value )
```

Setter for [nodeActivationNeedsDoubleClick\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.95 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.33.2.96 setRowsResizable()

```
setRowsResizable (
    value )
```

Setter for [rowsResizable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.97 setShowChangeMenu()

```
setShowChangeMenu (
    value )
```

Setter for [showChangeMenu\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.98 setShowOnlyFirstColumn()

```
setShowOnlyFirstColumn (
    value )
```

Setter for [showOnlyFirstColumn\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.99 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.100 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.101 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.102 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.103 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.33.2.104 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.105 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.106 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.33.2.107 showChangeMenu()

```
showChangeMenu ( )
```

If a menu shall be shown for making manual changes to the data source.

Instead of true, it may also be a configuration object, which may contain the following:

- `item_foo_label`, `item_foo_icon`, `item_foo_isvisible`, `item_foo_action`: Specifies a menu action `foo` by four functions. Each function gets `widget`, `datasource` as parameters. Respectively they have to return a label, an icon, if it is actually visible, or have to execute the action. It is also possible to customize the existing actions `addrow`, `addrowbefore`, `addcolumn`, `addcolumnbefore`, `removerow`, `removecolumn` and edit this way.

1.33.2.108 showOnlyFirstColumn()

```
showOnlyFirstColumn ( )
```

If to show only the first column and hide the other ones.

1.33.2.109 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretch←Affinity() == 0`.

1.33.2.110 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.33.2.111 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.33.2.112 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.33.2.113 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.33.2.114 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.33.2.115 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.33.2.116 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

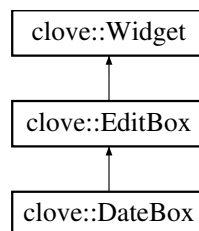
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.34 clove::DateBox Class Reference

A user input box for date input.

Inheritance diagram for clove::DateBox:



Public Member Functions

- [date](#) ()
The date value in the box as JavaScript Date.
- [setDate](#) (value)
Setter for [date\(\)](#).
- [readOnly](#) ()
If the edit box is read-only or editable by the user.
- [setReadOnly](#) (value)
Setter for [readOnly\(\)](#).
- [text](#) ()
The current text.
- [setText](#) (value)
Setter for [text\(\)](#).
- [hintText](#) ()

- The hint text.

 - `setHintText` (value)

Setter for `hintText()`.
 - `autocompleteItems` ()

A [Clove.DataSource](#) with a list of autocomplete items to popup.
 - `setAutocompleteItems` (value)

Setter for `autocompleteItems()`.
 - `autocompleteFilter` ()

How to filter the autocomplete list.
 - `setAutocompleteFilter` (value)

Setter for `autocompleteFilter()`.
 - `autocompleteOpenForNoText` ()

If to show the autocomplete list when the edit box is empty.
 - `setAutocompleteOpenForNoText` (value)

Setter for `autocompleteOpenForNoText()`.
 - `setTextSelection` (ifrom, ito)

Selects the specified part of the text.
 - `textSelection` ()

Returns the current text selection indices.
 - `OnChanged`

Triggered when the text was changed.
 - `OnPopupTextSelected`

Triggered when a text was selected from the popup.
 - `declareProperty` (k, defaultV)

Declares a widget property.
 - `getProperty` (k)

General-purpose getter for widget properties.
 - `setProperty` (k, v)

General-purpose setter for widget properties.
 - `bindProperty` (k, vb)

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
 - `init` (rootNameScope)

Initializes the widget.
 - `doinit` ()

Executes late widget initialization (i.e. after properties are applied).
 - `doinitEarly` ()

Executes early widget initialization (i.e. before properties are applied).
 - `resize` ()

Applies the new widget size to internal content.
 - `doresize` ()

Corrects alignments of internal elements according to the new widget size.
 - `relayout` ()

Notifies the parent widget that a new geometry is required.
 - `focus` ()

Sets the focus to this widget.
 - `isAlive` ()

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
 - `computeMinimalWidth` ()

Computes the minimal width in pixel this widget needs to have.
 - `computePreferredWidth` ()

Computes the preferred width in pixel this widget needs to have.

- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (class)
Adds the css class `class` to this widget.
- [removeStyleClass](#) (class)
Removes the css class `class` from this widget.
- [isStyleClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()

- If this widget is visible.*

 - `setVisibility` (value)
 Setter for `visibility()`.
- `doStandaloneResizing` ()
If the widget handles to resize itself as needed.

 - `setDoStandaloneResizing` (value)
 Setter for `doStandaloneResizing()`.
- `mayFocus` ()
If the widget can have the keyboard focus.

 - `setMayFocus` (value)
 Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

 - `setHorizontalStretchAffinity` (value)
 Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

 - `setVerticalStretchAffinity` (value)
 Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing` ()
If the widget is strictly forbidden to get additional horizontal space.

 - `setStrictHorizontalSizing` (value)
 Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing` ()
If the widget is strictly forbidden to get additional vertical space.

 - `setStrictVerticalSizing` (value)
 Setter for `strictVerticalSizing()`.
- `vstretch` ()
Alias for `verticalStretchAffinity()`.

 - `setVstretch` (value)
 Setter for `vstretch()`.
- `hstretch` ()
Alias for `horizontalStretchAffinity()`.

 - `setHstretch` (value)
 Setter for `hstretch()`.
- `busy` ()
If the widget is in busy state, typically resulting in a loading animation.

 - `setBusy` (value)
 Setter for `busy()`.
- `registerBusy` ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy` (token)
Drops a token and returns to normal state if no other tokens exist.
- `hasFocus` ()
If the widget has the keyboard focus.
- `innerSize` ()
Returns inner width and height of this widget.
- `outerSize` ()
Returns outer width and height of this widget.
- `OnDestroyed`
Triggered when this widget was removed somehow.

- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.34.1 Detailed Description

A user input box for date input.

The actual behavior depends on the browser.

1.34.2 Member Function Documentation

1.34.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.34.2.2 autoCompleteFilter()

```
autoCompleteFilter ( ) [inherited]
```

How to filter the autocomplete list.

Either `'startswith', 'contains', function(itemtext, boxtext)` or undefined.

1.34.2.3 autoCompleteItems()

```
autoCompleteItems ( ) [inherited]
```

A [Clove.DataSource](#) with a list of autocomplete items to popup.

It is allowed to observe the `text` in order to generate live content for this list.

1.34.2.4 autoCompleteOpenForNoText()

```
autoCompleteOpenForNoText ( ) [inherited]
```

If to show the autocomplete list when the edit box is empty.

1.34.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The Clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.34.2.6 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.34.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [Clove::Widget](#) instances.

1.34.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.34.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.34.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.34.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.34.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.34.2.13 date()

```
date ( )
```

The date value in the box as JavaScript Date.

1.34.2.14 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.34.2.15 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.34.2.16 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.34.2.17 `doresize()`

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.34.2.18 `doStandaloneResizing()`

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.34.2.19 `effectivelyEnabled()`

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.34.2.20 `effectiveVisibility()`

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.34.2.21 `enabled()`

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.34.2.22 `focus()`

`focus () [inherited]`

Sets the focus to this widget.

1.34.2.23 `getMinimalHeightForWidth()`

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.34.2.24 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.34.2.25 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.34.2.26 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.34.2.27 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.34.2.28 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.34.2.29 hintText()

```
hintText ( ) [inherited]
```

The hint text.

Typically contains something like a label text. It is shown as a hint when the box is empty.

1.34.2.30 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.34.2.31 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.34.2.32 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.34.2.33 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.34.2.34 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.34.2.35 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.34.2.36 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.34.2.37 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.34.2.38 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.34.2.39 OnChanged()

`OnChanged [inherited]`

Triggered when the text was changed.

This is a [clove.Event](#) instance. See Manual for details.

1.34.2.40 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.34.2.41 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.34.2.42 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.34.2.43 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.34.2.44 OnPopupTextSelected()

OnPopupTextSelected [inherited]

Triggered when a text was selected from the popup.

This is a [clove.Event](#) instance. See Manual for details.

1.34.2.45 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.34.2.46 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.34.2.47 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.34.2.48 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.34.2.49 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.34.2.50 readOnly()

```
readOnly ( ) [inherited]
```

If the edit box is read-only or editable by the user.

1.34.2.51 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.34.2.52 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.34.2.53 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.34.2.54 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.34.2.55 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.34.2.56 `setAutocompletionFilter()`

```
setAutocompletionFilter (
    value ) [inherited]
```

Setter for [autocompletionFilter\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.57 `setAutocompletionItems()`

```
setAutocompletionItems (
    value ) [inherited]
```

Setter for [autocompletionItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.58 `setAutocompletionOpenForNoText()`

```
setAutocompletionOpenForNoText (
    value ) [inherited]
```

Setter for [autocompleteOpenForNoText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.59 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.60 `setDate()`

```
setDate (
    value )
```

Setter for [date\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.61 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.62 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.63 setHintText()

```
setHintText (
    value ) [inherited]
```

Setter for [hintText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.64 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.65 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.66 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.67 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.68 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.34.2.69 setReadOnly()

```
setReadOnly (
    value ) [inherited]
```

Setter for [readOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.70 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.71 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.72 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.73 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.74 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.34.2.75 `setText()`

```
setText (
    value ) [inherited]
```

Setter for [text\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.76 setTextSelection()

```
setTextSelection (
    ifrom,
    ito ) [inherited]
```

Selects the specified part of the text.

Parameters

<i>ifrom</i>	The selection begin index.
<i>ito</i>	The selection end index.

1.34.2.77 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.78 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.79 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.34.2.80 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.34.2.81 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.34.2.82 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.34.2.83 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.34.2.84 text()

```
text ( ) [inherited]
```

The current text.

1.34.2.85 textSelection()

```
textSelection ( ) [inherited]
```

Returns the current text selection indices.

1.34.2.86 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.34.2.87 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.34.2.88 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.34.2.89 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

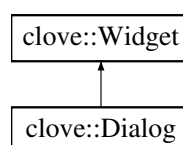
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.35 clove::Dialog Class Reference

A dialog is a container for a new sub user interface, decorated with a border and a title bar.

Inheritance diagram for `clove::Dialog`:



Public Member Functions

- **body** ()
The inner widget as widget configuration like for [clove::build\(\)](#).
- **setBody** (value)
Setter for [body\(\)](#).
- **title** ()
The dialog title text.
- **setTitle** (value)
Setter for [title\(\)](#).
- **titleicon** ()
The dialog title icon as [clove::Icon](#).
- **setTitleicon** (value)
Setter for [titleicon\(\)](#).
- **close** ()
Closes and removes this dialog.
- static **show** (dlg, showconfig)
Shows a dialog.
- **declareProperty** (k, defaultV)
Declares a widget property.
- **getProperty** (k)
General-purpose getter for widget properties.
- **setProperty** (k, v)
General-purpose setter for widget properties.
- **bindProperty** (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- **init** (rootNameScope)
Initializes the widget.
- **doinit** ()
Executes late widget initialization (i.e. after properties are applied).
- **doinitEarly** ()
Executes early widget initialization (i.e. before properties are applied).
- **resize** ()
Applies the new widget size to internal content.
- **doresize** ()
Corrects alignments of internal elements according to the new widget size.
- **relayout** ()
Notifies the parent widget that a new geometry is required.
- **focus** ()
Sets the focus to this widget.
- **isAlive** ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- **computeMinimalWidth** ()
Computes the minimal width in pixel this widget needs to have.
- **computePreferredWidth** ()
Computes the preferred width in pixel this widget needs to have.
- **computeMinimalHeightForWidth** (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- **computePreferredHeightForWidth** (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- **getMinimalWidth** ()

- Returns the minimal width in pixel this widget needs to have.*

 - [getPreferredWidth](#) ()

Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (clss)

Adds the css class `clss` to this widget.
- [removeStyleClass](#) (clss)

Removes the css class `clss` from this widget.
- [isStyleClass](#) (clss)

Checks if this widget contains the css class `clss`.
- [setStyleClassAssigned](#) (clss, assigned)

Sets or unsets the css class `clss` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.
- [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)

Removes this widget.
- [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()

List of the children `clove::Widget` instances.
- [parentWidget](#) ()

The parent `clove::Widget`.
- [name](#) ()

The name of the widget.
- [setName](#) (value)

Setter for [name\(\)](#).
- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)

Setter for [enabled\(\)](#).
- [styleClass](#) ()

Custom css class(es).
- [setStyleClass](#) (value)

Setter for [styleClass\(\)](#).
- [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)

Setter for [style\(\)](#).
- [visibility](#) ()

If this widget is visible.
- [setVisibility](#) (value)

Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()

If the widget handles to resize itself as needed.

- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)

Triggered whenever a property of this widget changed its value.

- [OnKeyDown](#)

Triggered when a keyboard key went down.

- [OnKeyUp](#)

Triggered when a keyboard key came up.

- [OnKeyPress](#)

Triggered when a keyboard key was pressed.

1.35.1 Detailed Description

A dialog is a container for a new sub user interface, decorated with a border and a title bar.

1.35.2 Member Function Documentation

1.35.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.35.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<code>k</code>	The widget property name.
<code>vb</code>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.35.2.3 body()

```
body ( )
```

The inner widget as widget configuration like for [clove::build\(\)](#).

1.35.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.35.2.5 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.35.2.6 close()

```
close ( )
```

Closes and removes this dialog.

1.35.2.7 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.35.2.8 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.35.2.9 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.35.2.10 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.35.2.11 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.35.2.12 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.35.2.13 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.35.2.14 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.35.2.15 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.35.2.16 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.35.2.17 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.35.2.18 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.35.2.19 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.35.2.20 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.35.2.21 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.35.2.22 `getMinimalWidth()`

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.35.2.23 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.35.2.24 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.35.2.25 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.35.2.26 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.35.2.27 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.35.2.28 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.35.2.29 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.35.2.30 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.35.2.31 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.35.2.32 `isStyleClass()`

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.35.2.33 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.35.2.34 `name()`

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.35.2.35 `nameScope()`

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.35.2.36 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.35.2.37 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.35.2.38 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.35.2.39 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.35.2.40 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.35.2.41 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.35.2.42 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [Clove.Event](#) instance. See Manual for details.

1.35.2.43 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.35.2.44 parentWidget()

parentWidget () [inherited]

The parent [Clove::Widget](#).

1.35.2.45 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.35.2.46 relayout()

relayout () [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.35.2.47 remove()

remove (
 removeconfig) [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.35.2.48 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.35.2.49 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.35.2.50 setBody()

```
setBody (
    value )
```

Setter for [body\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.51 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.52 setDoStandaloneResizing()

```
setDoStandaloneResizing (  
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.53 setEnabled()

```
setEnabled (  
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.54 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (  
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.55 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.56 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.57 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.58 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.35.2.59 `setStrictHorizontalSizing()`

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.60 `setStrictVerticalSizing()`

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.61 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.62 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.63 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.35.2.64 setTitle()

```
setTitle (
    value )
```

Setter for [title\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.65 setTitleicon()

```
setTitleicon (
    value )
```

Setter for [titleicon\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.66 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (  
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.67 setVisibility()

```
setVisibility (  
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.68 setVstretch()

```
setVstretch (  
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.35.2.69 show()

```
static show (
    dlg,
    showconfig )
```

Shows a dialog.

The root view of `dlg` is expected to be a [clove::Dialog](#) (or a subclass).

Parameters

<i>dlg</i>	The widget configuration, as for clove::build() , which specifies the dialog.
<i>showconfig</i>	A configuration object which specifies some presentation aspects.

1.35.2.70 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.35.2.71 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.35.2.72 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.35.2.73 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.35.2.74 title()

```
title ( )
```

The dialog title text.

1.35.2.75 titleicon()

```
titleicon ( )
```

The dialog title icon as [clove::Icon](#).

1.35.2.76 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.35.2.77 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.35.2.78 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.35.2.79 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

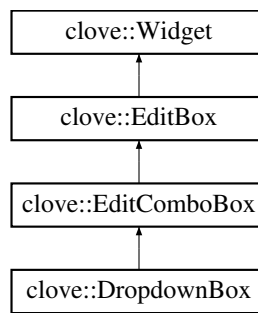
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.36 clove::DropDownBox Class Reference

A single-line text box (without edit functionality) with a popup list of values to select from.

Inheritance diagram for clove::DropDownBox:



Public Member Functions

- [popupItems](#) ()
A [clove.DataSource](#) with a list of items for the popup.
- [setPopupItems](#) (value)
Setter for [popupItems\(\)](#).
- [readOnly](#) ()
If the edit box is read-only or editable by the user.
- [setReadOnly](#) (value)
Setter for [readOnly\(\)](#).
- [text](#) ()
The current text.
- [setText](#) (value)
Setter for [text\(\)](#).
- [hintText](#) ()
The hint text.
- [setHintText](#) (value)
Setter for [hintText\(\)](#).
- [autocompleteItems](#) ()
A [clove.DataSource](#) with a list of autocomplete items to popup.
- [setAutocompleteItems](#) (value)
Setter for [autocompleteItems\(\)](#).
- [autocompleteFilter](#) ()

- How to filter the autocompletion list.*

 - `setAutocompletionFilter` (value)
Setter for `autocompletionFilter()`.
- `autocompletionOpenForNoText` ()
If to show the autocompletion list when the edit box is empty.
- `setAutocompletionOpenForNoText` (value)
Setter for `autocompletionOpenForNoText()`.
- `setTextSelection` (ifrom, ito)
Selects the specified part of the text.
- `textSelection` ()
Returns the current text selection indices.
- `OnChanged`
Triggered when the text was changed.
- `OnPopupTextSelected`
Triggered when a text was selected from the popup.
- `declareProperty` (k, defaultV)
Declares a widget property.
- `getProperty` (k)
General-purpose getter for widget properties.
- `setProperty` (k, v)
General-purpose setter for widget properties.
- `bindProperty` (k, vb)
Binds a `DataBinding` to a property. Read Manual for details about data bindings.
- `init` (rootNameScope)
Initializes the widget.
- `doinit` ()
Executes late widget initialization (i.e. after properties are applied).
- `doinitEarly` ()
Executes early widget initialization (i.e. before properties are applied).
- `resize` ()
Applies the new widget size to internal content.
- `doresize` ()
Corrects alignments of internal elements according to the new widget size.
- `relayout` ()
Notifies the parent widget that a new geometry is required.
- `focus` ()
Sets the focus to this widget.
- `isAlive` ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- `computeMinimalWidth` ()
Computes the minimal width in pixel this widget needs to have.
- `computePreferredWidth` ()
Computes the preferred width in pixel this widget needs to have.
- `computeMinimalHeightForWidth` (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- `computePreferredHeightForWidth` (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- `getMinimalWidth` ()
Returns the minimal width in pixel this widget needs to have.
- `getPreferredWidth` ()
Returns the preferred width in pixel this widget needs to have.

- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (clss)
Adds the css class `clss` to this widget.
- [removeStyleClass](#) (clss)
Removes the css class `clss` from this widget.
- [isStyleClass](#) (clss)
Checks if this widget contains the css class `clss`.
- [setStyleClassAssigned](#) (clss, assigned)
Sets or unsets the css class `clss` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()

- If the widget can have the keyboard focus.*

 - **setMayFocus** (value)
Setter for `mayFocus()`.
- **horizontalStretchAffinity** ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- **setHorizontalStretchAffinity** (value)
Setter for `horizontalStretchAffinity()`.
- **verticalStretchAffinity** ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- **setVerticalStretchAffinity** (value)
Setter for `verticalStretchAffinity()`.
- **strictHorizontalSizing** ()

If the widget is strictly forbidden to get additional horizontal space.
- **setStrictHorizontalSizing** (value)
Setter for `strictHorizontalSizing()`.
- **strictVerticalSizing** ()

If the widget is strictly forbidden to get additional vertical space.
- **setStrictVerticalSizing** (value)
Setter for `strictVerticalSizing()`.
- **vstretch** ()
Alias for `verticalStretchAffinity()`.
- **setVstretch** (value)
Setter for `vstretch()`.
- **hstretch** ()
Alias for `horizontalStretchAffinity()`.
- **setHstretch** (value)
Setter for `hstretch()`.
- **busy** ()

If the widget is in busy state, typically resulting in a loading animation.
- **setBusy** (value)
Setter for `busy()`.
- **registerBusy** ()

Sets the widget to busy state and returns a token which helps returning to normal state.
- **unregisterBusy** (token)

Drops a token and returns to normal state if no other tokens exist.
- **hasFocus** ()

If the widget has the keyboard focus.
- **innerSize** ()

Returns inner width and height of this widget.
- **outerSize** ()

Returns outer width and height of this widget.
- **OnDestroyed**

Triggered when this widget was removed somehow.
- **OnResized**

Triggered when this widget was resized.
- **OnVisibilityChanged**

Triggered when the visibility of this widget changed.
- **OnPropertyChanged**

Triggered whenever a property of this widget changed its value.
- **OnKeyDown**

Triggered when a keyboard key went down.

- [OnKeyUp](#)

Triggered when a keyboard key came up.

- [OnKeyPress](#)

Triggered when a keyboard key was pressed.

1.36.1 Detailed Description

A single-line text box (without edit functionality) with a popup list of values to select from.

1.36.2 Member Function Documentation

1.36.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.36.2.2 autoCompleteFilter()

```
autoCompleteFilter ( ) [inherited]
```

How to filter the autocomplete list.

Either `'startswith', 'contains', function(itemtext, boxtext)` or undefined.

1.36.2.3 autoCompleteItems()

```
autoCompleteItems ( ) [inherited]
```

A [clove.DataSource](#) with a list of autocomplete items to popup.

It is allowed to observe the `text` in order to generate live content for this list.

1.36.2.4 autoCompleteOpenForNoText()

```
autoCompleteOpenForNoText ( ) [inherited]
```

If to show the autocomplete list when the edit box is empty.

1.36.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.36.2.6 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.36.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.36.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.36.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.36.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.36.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.36.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.36.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.36.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.36.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.36.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.36.2.17 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.36.2.18 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.36.2.19 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.36.2.20 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.36.2.21 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.36.2.22 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.36.2.23 `getMinimalWidth()`

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.36.2.24 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.36.2.25 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.36.2.26 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.36.2.27 hasFocus()

`hasFocus () [inherited]`

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.36.2.28 hintText()

`hintText () [inherited]`

The hint text.

Typically contains something like a label text. It is shown as a hint when the box is empty.

1.36.2.29 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.36.2.30 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.36.2.31 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.36.2.32 `innerSize()`

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.36.2.33 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.36.2.34 `isStyleClass()`

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.36.2.35 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.36.2.36 `name()`

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.36.2.37 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.36.2.38 OnChanged()

`OnChanged [inherited]`

Triggered when the text was changed.

This is a [clove.Event](#) instance. See Manual for details.

1.36.2.39 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.36.2.40 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.36.2.41 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.36.2.42 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.36.2.43 OnPopupTextSelected()

OnPopupTextSelected [inherited]

Triggered when a text was selected from the popup.

This is a [clove.Event](#) instance. See Manual for details.

1.36.2.44 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.36.2.45 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.36.2.46 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.36.2.47 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.36.2.48 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.36.2.49 `popupItems()`

```
popupItems ( ) [inherited]
```

A [clove.DataSource](#) with a list of items for the popup.

1.36.2.50 `readOnly()`

```
readOnly ( ) [inherited]
```

If the edit box is read-only or editable by the user.

1.36.2.51 `registerBusy()`

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.36.2.52 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.36.2.53 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.36.2.54 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.36.2.55 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.36.2.56 `setAutocompletionFilter()`

```
setAutocompletionFilter (
    value ) [inherited]
```

Setter for [autocompletionFilter\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.57 `setAutocompletionItems()`

```
setAutocompletionItems (
    value ) [inherited]
```

Setter for [autocompletionItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.58 setAutocompletionOpenForNoText()

```
setAutocompletionOpenForNoText (
    value ) [inherited]
```

Setter for [autocompletionOpenForNoText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.59 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.60 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.61 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.62 `setHintText()`

```
setHintText (
    value ) [inherited]
```

Setter for [hintText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.63 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.64 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.65 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.66 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.67 setPopuItems()

```
setPopuItems (
    value ) [inherited]
```

Setter for [popuItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.68 `setProperty()`

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.36.2.69 `setReadOnly()`

```
setReadOnly (
    value ) [inherited]
```

Setter for [readOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.70 `setStrictHorizontalSizing()`

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.71 `setStrictVerticalSizing()`

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.72 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.73 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.74 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.36.2.75 setText()

```
setText (
    value ) [inherited]
```

Setter for [text\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.76 setTextSelection()

```
setTextSelection (
    ifrom,
    ito ) [inherited]
```

Selects the specified part of the text.

Parameters

<i>ifrom</i>	The selection begin index.
<i>ito</i>	The selection end index.

1.36.2.77 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.78 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.79 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.36.2.80 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.36.2.81 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.36.2.82 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.36.2.83 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.36.2.84 text()

```
text ( ) [inherited]
```

The current text.

1.36.2.85 textSelection()

```
textSelection ( ) [inherited]
```

Returns the current text selection indices.

1.36.2.86 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.36.2.87 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.36.2.88 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.36.2.89 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

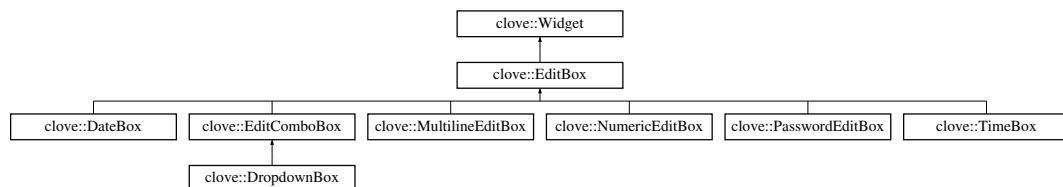
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.37 clove::EditBox Class Reference

A single-line box for text input from the user.

Inheritance diagram for clove::EditBox:



Public Member Functions

- [readOnly\(\)](#)
If the edit box is read-only or editable by the user.
- [setReadOnly\(\)](#) (value)
Setter for [readOnly\(\)](#).
- [text\(\)](#)
The current text.
- [setText\(\)](#) (value)
Setter for [text\(\)](#).
- [hintText\(\)](#)
The hint text.
- [setHintText\(\)](#) (value)
Setter for [hintText\(\)](#).
- [autocompleteItems\(\)](#)
A [clove::Datasource](#) with a list of autocomplete items to popup.

- [setAutocompletionItems](#) (value)
Setter for [autocompletionItems\(\)](#).
- [autocompletionFilter](#) ()
How to filter the autocompletion list.
- [setAutocompletionFilter](#) (value)
Setter for [autocompletionFilter\(\)](#).
- [autocompletionOpenForNoText](#) ()
If to show the autocompletion list when the edit box is empty.
- [setAutocompletionOpenForNoText](#) (value)
Setter for [autocompletionOpenForNoText\(\)](#).
- [setTextSelection](#) (ifrom, ito)
Selects the specified part of the text.
- [textSelection](#) ()
Returns the current text selection indices.
- [OnChanged](#)
Triggered when the text was changed.
- [OnPopupTextSelected](#)
Triggered when a text was selected from the popup.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()

- Returns the minimal width in pixel this widget needs to have.*

 - [getPreferredWidth](#) ()

Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (clss)

Adds the css class `clss` to this widget.
- [removeStyleClass](#) (clss)

Removes the css class `clss` from this widget.
- [isStyleClass](#) (clss)

Checks if this widget contains the css class `clss`.
- [setStyleClassAssigned](#) (clss, assigned)

Sets or unsets the css class `clss` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.
- [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)

Removes this widget.
- [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()

List of the children `clove::Widget` instances.
- [parentWidget](#) ()

The parent `clove::Widget`.
- [name](#) ()

The name of the widget.
- [setName](#) (value)

Setter for [name\(\)](#).
- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)

Setter for [enabled\(\)](#).
- [styleClass](#) ()

Custom css class(es).
- [setStyleClass](#) (value)

Setter for [styleClass\(\)](#).
- [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)

Setter for [style\(\)](#).
- [visibility](#) ()

If this widget is visible.
- [setVisibility](#) (value)

Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()

If the widget handles to resize itself as needed.

- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)

Triggered whenever a property of this widget changed its value.

- [OnKeyDown](#)

Triggered when a keyboard key went down.

- [OnKeyUp](#)

Triggered when a keyboard key came up.

- [OnKeyPress](#)

Triggered when a keyboard key was pressed.

1.37.1 Detailed Description

A single-line box for text input from the user.

1.37.2 Member Function Documentation

1.37.2.1 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.37.2.2 `autocompleteFilter()`

```
autocompleteFilter ( )
```

How to filter the autocomplete list.

Either `'startswith', 'contains', function(itemtext, boxtext)` or undefined.

1.37.2.3 `autocompleteItems()`

```
autocompleteItems ( )
```

A [clove.Datasource](#) with a list of autocomplete items to popup.

It is allowed to observe the `text` in order to generate live content for this list.

1.37.2.4 autoCompleteOpenForNoText()

```
autoCompletionOpenForNoText ( )
```

If to show the autocomplete list when the edit box is empty.

1.37.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.37.2.6 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.37.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.37.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.37.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.37.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.37.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.37.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.37.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.37.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.37.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.37.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.37.2.17 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.37.2.18 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.37.2.19 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.37.2.20 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.37.2.21 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.37.2.22 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.37.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.37.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.37.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.37.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.37.2.27 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.37.2.28 hintText()

```
hintText ( )
```

The hint text.

Typically contains something like a label text. It is shown as a hint when the box is empty.

1.37.2.29 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.37.2.30 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.37.2.31 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.37.2.32 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.37.2.33 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.37.2.34 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.37.2.35 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.37.2.36 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.37.2.37 nameScope()

nameScope () [inherited]

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.37.2.38 OnChanged()

OnChanged

Triggered when the text was changed.

This is a [clove.Event](#) instance. See Manual for details.

1.37.2.39 OnDestroyed()

OnDestroyed [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.37.2.40 OnKeyDown()

OnKeyDown [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.37.2.41 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.37.2.42 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.37.2.43 OnPopupTextSelected()

`OnPopupTextSelected`

Triggered when a text was selected from the popup.

This is a [clove.Event](#) instance. See Manual for details.

1.37.2.44 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.37.2.45 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.37.2.46 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.37.2.47 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.37.2.48 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.37.2.49 `readOnly()`

```
readOnly ( )
```

If the edit box is read-only or editable by the user.

1.37.2.50 `registerBusy()`

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.37.2.51 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.37.2.52 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.37.2.53 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.37.2.54 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.37.2.55 `setAutocompletionFilter()`

```
setAutocompletionFilter (
    value )
```

Setter for [autocompletionFilter\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.56 `setAutocompletionItems()`

```
setAutocompletionItems (
    value )
```

Setter for [autocompletionItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.57 `setAutocompletionOpenForNoText()`

```
setAutocompletionOpenForNoText (
    value )
```

Setter for [autocompleteOpenForNoText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.58 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.59 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.60 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.61 `setHintText()`

```
setHintText (
    value )
```

Setter for [hintText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.62 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.63 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.64 `setMayFocus()`

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.65 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.66 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.37.2.67 setReadOnly()

```
setReadOnly (
    value )
```

Setter for [readOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.68 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (  
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.69 setStrictVerticalSizing()

```
setStrictVerticalSizing (  
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.70 setStyle()

```
setStyle (  
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.71 setStyleClass()

```
setStyleClass (  
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.72 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.37.2.73 `setText()`

```
setText (
    value )
```

Setter for [text\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.74 `setTextSelection()`

```
setTextSelection (
    ifrom,
    ito )
```

Selects the specified part of the text.

Parameters

<i>ifrom</i>	The selection begin index.
<i>ito</i>	The selection end index.

1.37.2.75 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.76 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.77 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.37.2.78 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.37.2.79 `strictVerticalSizing()`

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity()==0`.

1.37.2.80 `style()`

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.37.2.81 `styleClass()`

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.37.2.82 `text()`

```
text ( )
```

The current text.

1.37.2.83 `textSelection()`

```
textSelection ( )
```

Returns the current text selection indices.

1.37.2.84 `unregisterBusy()`

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.37.2.85 `verticalStretchAffinity()`

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.37.2.86 `visibility()`

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.37.2.87 `vstretch()`

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

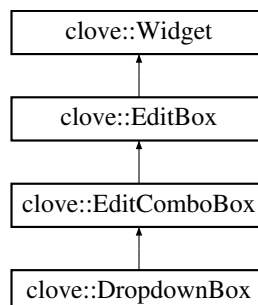
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.38 `clove::EditComboBox` Class Reference

A single-line box for text input from the user with an additional popup list of value recommendations.

Inheritance diagram for `clove::EditComboBox`:



Public Member Functions

- [popupItems](#) ()
A [clove.Datasource](#) with a list of items for the popup.
- [setPopupItems](#) (value)
Setter for [popupItems\(\)](#).
- [readOnly](#) ()
If the edit box is read-only or editable by the user.
- [setReadOnly](#) (value)
Setter for [readOnly\(\)](#).
- [text](#) ()
The current text.
- [setText](#) (value)
Setter for [text\(\)](#).
- [hintText](#) ()
The hint text.
- [setHintText](#) (value)
Setter for [hintText\(\)](#).
- [autocompletionItems](#) ()
A [clove.Datasource](#) with a list of autocompletion items to popup.
- [setAutocompletionItems](#) (value)
Setter for [autocompletionItems\(\)](#).
- [autocompletionFilter](#) ()
How to filter the autocompletion list.
- [setAutocompletionFilter](#) (value)
Setter for [autocompletionFilter\(\)](#).
- [autocompletionOpenForNoText](#) ()
If to show the autocompletion list when the edit box is empty.
- [setAutocompletionOpenForNoText](#) (value)
Setter for [autocompletionOpenForNoText\(\)](#).
- [setTextSelection](#) (ifrom, ito)
Selects the specified part of the text.
- [textSelection](#) ()
Returns the current text selection indices.
- [OnChanged](#)
Triggered when the text was changed.
- [OnPopupTextSelected](#)
Triggered when a text was selected from the popup.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read [Manual](#) for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()

- Executes early widget initialization (i.e. before properties are applied).*

 - `resize ()`
Applies the new widget size to internal content.
 - `doresize ()`
Corrects alignments of internal elements according to the new widget size.
 - `relayout ()`
Notifies the parent widget that a new geometry is required.
 - `focus ()`
Sets the focus to this widget.
 - `isAlive ()`
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
 - `computeMinimalWidth ()`
Computes the minimal width in pixel this widget needs to have.
 - `computePreferredWidth ()`
Computes the preferred width in pixel this widget needs to have.
 - `computeMinimalHeightForWidth (w)`
Computes the minimal height in pixel this widget needs to have for a given width.
 - `computePreferredHeightForWidth (w)`
Computes the preferred height in pixel this widget needs to have for a given width.
 - `getMinimalWidth ()`
Returns the minimal width in pixel this widget needs to have.
 - `getPreferredWidth ()`
Returns the preferred width in pixel this widget needs to have.
 - `getMinimalHeightForWidth (w)`
Returns the minimal height in pixel this widget needs to have for a given width.
 - `getPreferredHeightForWidth (w)`
Returns the preferred height in pixel this widget needs to have for a given width.
 - `addStyleClass (css)`
Adds the css class `css` to this widget.
 - `removeStyleClass (css)`
Removes the css class `css` from this widget.
 - `isStyleClass (css)`
Checks if this widget contains the css class `css`.
 - `setStyleClassAssigned (css, assigned)`
Sets or unsets the css class `css` to this widget.
 - `containingNameScope ()`
The namespace this widget contains to.
 - `nameScope ()`
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
 - `remove (removeconfig)`
Removes this widget.
 - `effectivelyEnabled ()`
If this widget is enabled and not marked as busy (i.e. can interact with the user).
 - `effectiveVisibility ()`
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
 - `childrenWidgets ()`
List of the children `clove::Widget` instances.
 - `parentWidget ()`
The parent `clove::Widget`.
 - `name ()`
The name of the widget.

- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()

- If the widget is in busy state, typically resulting in a loading animation.*

 - `setBusy` (value)
Setter for `busy()`.
 - `registerBusy` ()
Sets the widget to busy state and returns a token which helps returning to normal state.
 - `unregisterBusy` (token)
Drops a token and returns to normal state if no other tokens exist.
 - `hasFocus` ()
If the widget has the keyboard focus.
 - `innerSize` ()
Returns inner width and height of this widget.
 - `outerSize` ()
Returns outer width and height of this widget.
 - `OnDestroyed`
Triggered when this widget was removed somehow.
 - `OnResized`
Triggered when this widget was resized.
 - `OnVisibilityChanged`
Triggered when the visibility of this widget changed.
 - `OnPropertyChanged`
Triggered whenever a property of this widget changed its value.
 - `OnKeyDown`
Triggered when a keyboard key went down.
 - `OnKeyUp`
Triggered when a keyboard key came up.
 - `OnKeyPress`
Triggered when a keyboard key was pressed.

1.38.1 Detailed Description

A single-line box for text input from the user with an additional popup list of value recommendations.

1.38.2 Member Function Documentation

1.38.2.1 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.38.2.2 `autocompletionFilter()`

`autocompletionFilter ()` [inherited]

How to filter the autocompletion list.

Either `'startswith'`, `'contains'`, `function(itemtext, boxtext)` or `undefined`.

1.38.2.3 `autocompletionItems()`

`autocompletionItems ()` [inherited]

A [clove.DataSource](#) with a list of autocompletion items to popup.

It is allowed to observe the `text` in order to generate live content for this list.

1.38.2.4 `autocompletionOpenForNoText()`

`autocompletionOpenForNoText ()` [inherited]

If to show the autocompletion list when the edit box is empty.

1.38.2.5 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<code>k</code>	The widget property name.
<code>vb</code>	The clove.DataBinding to bind. May be <code>'undefined'</code> for just unbinding.

1.38.2.6 `busy()`

`busy ()` [inherited]

If the widget is in busy state, typically resulting in a loading animation.

1.38.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [Clove::Widget](#) instances.

1.38.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.38.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.38.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.38.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```


Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.38.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.38.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.38.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.38.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.38.2.16 `doresize()`

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.38.2.17 `doStandaloneResizing()`

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.38.2.18 `effectivelyEnabled()`

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.38.2.19 `effectiveVisibility()`

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.38.2.20 `enabled()`

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.38.2.21 `focus()`

`focus () [inherited]`

Sets the focus to this widget.

1.38.2.22 `getMinimalHeightForWidth()`

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.38.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.38.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.38.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.38.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.38.2.27 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.38.2.28 hintText()

```
hintText ( ) [inherited]
```

The hint text.

Typically contains something like a label text. It is shown as a hint when the box is empty.

1.38.2.29 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.38.2.30 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.38.2.31 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.38.2.32 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.38.2.33 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.38.2.34 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class *class*.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.38.2.35 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.38.2.36 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.38.2.37 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.38.2.38 OnChanged()

`OnChanged [inherited]`

Triggered when the text was changed.

This is a [clove.Event](#) instance. See Manual for details.

1.38.2.39 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.38.2.40 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.38.2.41 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.38.2.42 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.38.2.43 OnPopupTextSelected()

OnPopupTextSelected [inherited]

Triggered when a text was selected from the popup.

This is a [clove.Event](#) instance. See Manual for details.

1.38.2.44 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.38.2.45 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.38.2.46 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.38.2.47 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.38.2.48 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.38.2.49 `popupItems()`

```
popupItems ( )
```

A [Clove.Datasource](#) with a list of items for the popup.

1.38.2.50 `readOnly()`

```
readOnly ( ) [inherited]
```

If the edit box is read-only or editable by the user.

1.38.2.51 `registerBusy()`

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.38.2.52 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.38.2.53 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.38.2.54 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.38.2.55 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.38.2.56 setAutocompletionFilter()

```
setAutocompletionFilter (
    value ) [inherited]
```

Setter for [autocompletionFilter\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.57 setAutocompletionItems()

```
setAutocompletionItems (
    value ) [inherited]
```

Setter for [autocompletionItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.58 setAutocompletionOpenForNoText()

```
setAutocompletionOpenForNoText (
    value ) [inherited]
```

Setter for [autocompletionOpenForNoText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.59 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.60 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.61 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.62 `setHintText()`

```
setHintText (
    value ) [inherited]
```

Setter for [hintText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.63 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.64 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.65 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.66 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.67 setPopUpItems()

```
setPopUpItems (
    value )
```

Setter for [popupItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.68 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.38.2.69 setReadOnly()

```
setReadOnly (
    value ) [inherited]
```

Setter for [readOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.70 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.71 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.72 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.73 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.74 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.38.2.75 setText()

```
setText (
    value ) [inherited]
```

Setter for [text\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.76 setTextSelection()

```
setTextSelection (
    ifrom,
    ito ) [inherited]
```

Selects the specified part of the text.

Parameters

<i>ifrom</i>	The selection begin index.
<i>ito</i>	The selection end index.

1.38.2.77 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.78 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.79 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.38.2.80 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.38.2.81 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.38.2.82 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.38.2.83 styleClass()

`styleClass () [inherited]`

Custom css class(es).

1.38.2.84 text()

`text () [inherited]`

The current text.

1.38.2.85 textSelection()

`textSelection () [inherited]`

Returns the current text selection indices.

1.38.2.86 unregisterBusy()

`unregisterBusy (
 token) [inherited]`

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.38.2.87 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.38.2.88 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.38.2.89 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.39 clove::Event Class Reference

An event. It can have some handlers (or: listeners) and can be triggered.

Public Member Functions

- [Event](#) ()
- [trigger](#) (params)
Triggers the event.
- [addHandler](#) (fct, owner)
Adds a handler (or: listener) to this event.
- [hasHandlers](#) ()
Checks if this event has any handlers (otherwise triggering it doesn't have an effect).
- [addHandlerOnce](#) (fct, owner)
Like [addHandler\(\)](#), but automatically removes the handler after the event occurred once.
- [removeHandler](#) (fct)
Removes a handler from this event.

1.39.1 Detailed Description

An event. It can have some handlers (or: listeners) and can be triggered.

Read the Manual for details about the Clove event mechanism.

1.39.2 Constructor & Destructor Documentation

1.39.2.1 Event()

`Event ()`

1.39.3 Member Function Documentation

1.39.3.1 addHandler()

```
addHandler (
    fct,
    owner )
```

Adds a handler (or: listener) to this event.

Whenever the event is triggered afterwards, the new handler is called.

Parameters

<i>fct</i>	A handler function.
<i>owner</i>	Optional owner widget. If given, <code>removeHandler</code> will be called automatically after widget removal.

1.39.3.2 addHandlerOnce()

```
addHandlerOnce (
    fct,
    owner )
```

Like [addHandler\(\)](#), but automatically removes the handler after the event occurred once.

Parameters

<i>fct</i>	A handler function.
<i>owner</i>	Optional owner widget. If given, <code>removeHandler</code> will be called automatically after widget removal.

1.39.3.3 hasHandlers()

```
hasHandlers ( )
```

Checks if this event has any handlers (otherwise triggering it doesn't have an effect).

1.39.3.4 removeHandler()

```
removeHandler (
    fct )
```

Removes a handler from this event.

Parameters

<i>fct</i>	The handler function to remove.
------------	---------------------------------

1.39.3.5 trigger()

```
trigger (
    params )
```

Triggers the event.

This calls all registered handlers.

Parameters

<i>params</i>	Additional information about the event incidence.
---------------	---

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.40 clove::EventArgs Class Reference

Arguments for a particular incidence of a [clove::Event](#).

Public Member Functions

- [EventArgs](#) (params)
- [skipExecution](#) ()
Marks the further event handling for skipping.

1.40.1 Detailed Description

Arguments for a particular incidence of a [clove::Event](#).

It carries some additional informations about it. The details depend on the event (and the component which triggers it).

1.40.2 Constructor & Destructor Documentation

1.40.2.1 EventArgs()

```
EventArgs (
    params )
```

Parameters

<i>params</i>	Additional information about the event incidence.
---------------	---

1.40.3 Member Function Documentation

1.40.3.1 skipExecution()

```
skipExecution ( )
```

Marks the further event handling for skipping.

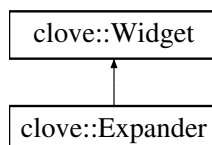
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.41 clove::Expander Class Reference

A single-widget container which can be expanded or collapsed.

Inheritance diagram for clove::Expander:



Public Member Functions

- [isExpanded](#) ()
If the expander is expanded.
- [setIsExpanded](#) (value)
Setter for [isExpanded\(\)](#).
- [collapsedIcon](#) ()
The icon for the expander when collapsed.
- [setCollapsedIcon](#) (value)
Setter for [collapsedIcon\(\)](#).
- [collapsedLabel](#) ()
The label for the expander when collapsed.
- [setCollapsedLabel](#) (value)
Setter for [collapsedLabel\(\)](#).
- [expandedIcon](#) ()
The icon for the expander when expanded.
- [setExpandedIcon](#) (value)
Setter for [expandedIcon\(\)](#).
- [expandedLabel](#) ()
The label for the expander when expanded.
- [setExpandedLabel](#) (value)
Setter for [expandedLabel\(\)](#).
- [body](#) ()
The inner [Clove::Widget](#).
- [setBody](#) (value)
Setter for [body\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()

- Computes the minimal width in pixel this widget needs to have.*

 - [computePreferredWidth](#) ()

Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)

Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)

Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()

Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()

Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (clss)

Adds the css class `clss` to this widget.
- [removeStyleClass](#) (clss)

Removes the css class `clss` from this widget.
- [isStyleClass](#) (clss)

Checks if this widget contains the css class `clss`.
- [setStyleClassAssigned](#) (clss, assigned)

Sets or unsets the css class `clss` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.
- [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)

Removes this widget.
- [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()

List of the children `clove::Widget` instances.
- [parentWidget](#) ()

The parent `clove::Widget`.
- [name](#) ()

The name of the widget.
- [setName](#) (value)

Setter for [name\(\)](#).
- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)

Setter for [enabled\(\)](#).
- [styleClass](#) ()

Custom css class(es).
- [setStyleClass](#) (value)

Setter for [styleClass\(\)](#).
- [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

- `setStyle` (value)
Setter for `style()`.
- `visibility` ()
If this widget is visible.
- `setVisibility` (value)
Setter for `visibility()`.
- `doStandaloneResizing` ()
If the widget handles to resize itself as needed.
- `setDoStandaloneResizing` (value)
Setter for `doStandaloneResizing()`.
- `mayFocus` ()
If the widget can have the keyboard focus.
- `setMayFocus` (value)
Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity` (value)
Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity` (value)
Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing` ()
If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing` (value)
Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing` ()
If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing` (value)
Setter for `strictVerticalSizing()`.
- `vstretch` ()
Alias for `verticalStretchAffinity()`.
- `setVstretch` (value)
Setter for `vstretch()`.
- `hstretch` ()
Alias for `horizontalStretchAffinity()`.
- `setHstretch` (value)
Setter for `hstretch()`.
- `busy` ()
If the widget is in busy state, typically resulting in a loading animation.
- `setBusy` (value)
Setter for `busy()`.
- `registerBusy` ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy` (token)
Drops a token and returns to normal state if no other tokens exist.
- `hasFocus` ()
If the widget has the keyboard focus.
- `innerSize` ()
Returns inner width and height of this widget.
- `outerSize` ()

- Returns outer width and height of this widget.*
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.41.1 Detailed Description

A single-widget container which can be expanded or collapsed.

1.41.2 Member Function Documentation

1.41.2.1 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.41.2.2 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.41.2.3 body()

```
body ( )
```

The inner [clove::Widget](#).

1.41.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.41.2.5 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.41.2.6 collapsedIcon()

```
collapsedIcon ( )
```

The icon for the expander when collapsed.

1.41.2.7 collapsedLabel()

```
collapsedLabel ( )
```

The label for the expander when collapsed.

1.41.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.41.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.41.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.41.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.41.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.41.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.41.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.41.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.41.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.41.2.17 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.41.2.18 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.41.2.19 effectiveVisibility()

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.41.2.20 enabled()

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.41.2.21 expandedIcon()

`expandedIcon ()`

The icon for the expander when expanded.

1.41.2.22 expandedLabel()

`expandedLabel ()`

The label for the expander when expanded.

1.41.2.23 focus()

`focus () [inherited]`

Sets the focus to this widget.

1.41.2.24 getMinimalHeightForWidth()

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.41.2.25 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.41.2.26 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.41.2.27 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.41.2.28 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.41.2.29 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.41.2.30 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.41.2.31 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.41.2.32 init()

```
init (   
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.41.2.33 `innerSize()`

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.41.2.34 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.41.2.35 `isExpanded()`

```
isExpanded ( )
```

If the expander is expanded.

1.41.2.36 `isStyleClass()`

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.41.2.37 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.41.2.38 `name()`

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.41.2.39 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.41.2.40 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.41.2.41 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.41.2.42 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.41.2.43 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.41.2.44 OnPropertyChanged()

`OnPropertyChanged [inherited]`

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.41.2.45 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [Clove.Event](#) instance. See Manual for details.

1.41.2.46 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [Clove.Event](#) instance. See Manual for details.

1.41.2.47 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.41.2.48 parentWidget()

`parentWidget ()` [inherited]

The parent [Clove::Widget](#).

1.41.2.49 registerBusy()

`registerBusy ()` [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.41.2.50 relayout()

`relayout ()` [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.41.2.51 remove()

`remove (`
 `removeconfig)` [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.41.2.52 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.41.2.53 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.41.2.54 setBody()

```
setBody (
    value )
```

Setter for [body\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.55 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.56 setCollapsedIcon()

```
setCollapsedIcon (  
    value )
```

Setter for [collapsedIcon\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.57 setCollapsedLabel()

```
setCollapsedLabel (  
    value )
```

Setter for [collapsedLabel\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.58 setDoStandaloneResizing()

```
setDoStandaloneResizing (  
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.59 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.60 setExpandedIcon()

```
setExpandedIcon (
    value )
```

Setter for [expandedIcon\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.61 setExpandedLabel()

```
setExpandedLabel (
    value )
```

Setter for [expandedLabel\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.62 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.63 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.64 setIsExpanded()

```
setIsExpanded (
    value )
```

Setter for [isExpanded\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.65 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.66 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.67 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.41.2.68 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.69 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.70 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.71 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.72 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.41.2.73 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.74 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.75 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.41.2.76 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.41.2.77 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.41.2.78 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.41.2.79 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.41.2.80 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.41.2.81 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.41.2.82 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.41.2.83 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

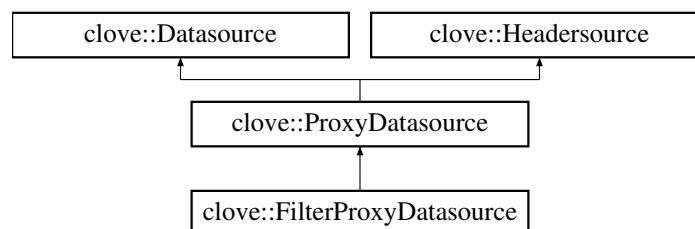
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.42 clove::FilterProxyDatasource Class Reference

A [clove::Datasource](#) implementation which filters another [clove::Datasource](#).

Inheritance diagram for [clove::FilterProxyDatasource](#):



Public Member Functions

- [FilterProxyDatasource](#) (datasource, rowfilterfct, colfilterfct)
- [setRowFilter](#) (rowfilterfct)
Sets the row filter function.
- [setColumnFilter](#) (colfilterfct)
Sets the column filter function.
- [setDatasource](#) (datasource)
Sets the source datasource.
- [proxyPointerToNativePointer](#) (proxyptr)
Translates a [clove::DatasourceValuePointer](#) from this proxy datasource to a one for the source datasource.
- [nativePointerToProxyPointer](#) (nativeptr)
Translates a [clove::DatasourceValuePointer](#) from the source datasource to a one for this proxy datasource.
- [refresh](#) ()
Refreshes the filtering.

- [getValue](#) (ptr)
Returns the value for a given node.
- [getMetadata](#) (ptr)
Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.
- [changeValue](#) (ptr, value)
Change the value for a given node.
- [rowCount](#) (parent)
Returns the number of rows for a given node.
- [columnCount](#) (parent)
Returns the number of columns for a given node.
- [valuePointer](#) (irow, icol, parent)
Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.
- [parent](#) (ptr)
Returns the parent node for a given node.
- [valuePointerNavigateInDepth](#) (ptr, direction, mayexpandfct)
Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.
- [OnDataInsert](#)
Triggered when a node insertion takes place.
- [OnDataRemove](#)
Triggered when a node removal takes place.
- [OnDataUpdate](#)
Triggered when a node data update takes place.
- [getRowHeader](#) (irow, parent)
Returns the header configuration for a given row.
- [getColumnHeader](#) (irow, parent)
Returns the header configuration for a given column.
- [rowHeadersVisible](#) (parent)
If row headers are visible.
- [columnHeadersVisible](#) (parent)
If column headers are visible.
- [OnHeaderDataInsert](#)
Triggered when a new row or column of header data was inserted.
- [OnHeaderDataRemove](#)
Triggered when a row or column of header data was removed.
- [OnHeaderDataUpdate](#)
Triggered when header data were updated.
- [OnHeaderVisibilityUpdated](#)
Triggered when header visibilities changed.

1.42.1 Detailed Description

A [clove::Datasource](#) implementation which filters another [clove::Datasource](#).

1.42.2 Constructor & Destructor Documentation

1.42.2.1 FilterProxyDatasource()

```
FilterProxyDatasource (
    datasource,
    rowfilterfct,
    colfilterfct )
```

Parameters

<i>datasource</i>	The source clove::Datasource .
<i>rowfilterfct</i>	A function(<i>datasource</i> , <i>idx</i> , <i>parent</i>) returning <code>true</code> if the filter accepts this row (optional).
<i>colfilterfct</i>	A function(<i>datasource</i> , <i>idx</i> , <i>parent</i>) returning <code>true</code> if the filter accepts this column (optional).

1.42.3 Member Function Documentation

1.42.3.1 `changeValue()`

```
changeValue (
    ptr,
    value ) [pure virtual], [inherited]
```

Change the value for a given node.

This method is called from external places, e.g. when a user makes changes in a datasource-connected widget.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>value</i>	The new value.

1.42.3.2 `columnCount()`

```
columnCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of columns for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.42.3.3 `columnHeadersVisible()`

```
columnHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If column headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.42.3.4 getColumnHeader()

```
getColumnHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given column.

Parameters

<i>irow</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.42.3.5 getMetadata()

```
getMetadata (
    ptr ) [pure virtual], [inherited]
```

Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.42.3.6 getRowHeader()

```
getRowHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given row.

Parameters

<i>irow</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.42.3.7 `getValue()`

```
getValue (
    ptr ) [pure virtual], [inherited]
```

Returns the value for a given node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.42.3.8 `nativePointerToProxyPointer()`

```
nativePointerToProxyPointer (
    nativeptr ) [inherited]
```

Translates a [clove::DatasourceValuePointer](#) from the source datasource to a one for this proxy datasource.

Parameters

<i>nativeptr</i>	A value pointer.
------------------	------------------

1.42.3.9 `OnDataInsert()`

```
OnDataInsert [inherited]
```

Triggered when a node insertion takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.42.3.10 `OnDataRemove()`

```
OnDataRemove [inherited]
```

Triggered when a node removal takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.42.3.11 OnDataUpdate()

OnDataUpdate [inherited]

Triggered when a node data update takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.42.3.12 OnHeaderDataInsert()

OnHeaderDataInsert [inherited]

Triggered when a new row or column of header data was inserted.

This is a [clove.Event](#) instance. See Manual for details.

1.42.3.13 OnHeaderDataRemove()

OnHeaderDataRemove [inherited]

Triggered when a row or column of header data was removed.

This is a [clove.Event](#) instance. See Manual for details.

1.42.3.14 OnHeaderDataUpdate()

OnHeaderDataUpdate [inherited]

Triggered when header data were updated.

This is a [clove.Event](#) instance. See Manual for details.

1.42.3.15 OnHeaderVisibilityUpdated()

OnHeaderVisibilityUpdated [inherited]

Triggered when header visibilities changed.

This is a [clove.Event](#) instance. See Manual for details.

1.42.3.16 parent()

```
parent (
    ptr ) [pure virtual], [inherited]
```

Returns the parent node for a given node.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
------------	---

1.42.3.17 proxyPointerToNativePointer()

```
proxyPointerToNativePointer (
    proxyptr ) [inherited]
```

Translates a [clove::DatasourceValuePointer](#) from this proxy datasource to a one for the source datasource.

Parameters

<i>proxyptr</i>	A value pointer.
-----------------	------------------

1.42.3.18 refresh()

```
refresh ( ) [inherited]
```

Refreshes the filtering.

Call this when the outer conditions changed and the filters must be applied again.

1.42.3.19 rowCount()

```
rowCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of rows for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.42.3.20 rowHeadersVisible()

```
rowHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If row headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.42.3.21 setColumnFilter()

```
setColumnFilter (
    colfilterfct )
```

Sets the column filter function.

Parameters

<i>colfilterfct</i>	The column filter function. See constructor for details.
---------------------	--

1.42.3.22 setDatasource()

```
setDatasource (
    datasource ) [inherited]
```

Sets the source datasource.

Parameters

<i>datasource</i>	The source clove::Datasource .
-------------------	--

1.42.3.23 setRowFilter()

```
setRowFilter (
    rowfilterfct )
```

Sets the row filter function.

Parameters

<i>rowfilterfct</i>	The row filter function. See constructor for details.
---------------------	---

1.42.3.24 valuePointer()

```
valuePointer (
    irow,
    icol,
    parent ) [pure virtual], [inherited]
```

Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.

Parameters

<i>irow</i>	The row index.
<i>icol</i>	The column index.
<i>parent</i>	The parent as clove::DatasourceValuePointer .

1.42.3.25 valuePointerNavigateInDepth()

```
valuePointerNavigateInDepth (
    ptr,
    direction,
    mayexpandfct ) [inherited]
```

Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
<i>direction</i>	The direction (+1 or -1).
<i>mayexpandfct</i>	A function(<i>ptr</i>) which returns <code>true</code> iff this node's children are to be traversed.

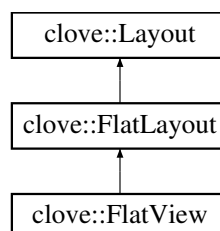
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.43 clove::FlatLayout Class Reference

A mixin for flat layout implementations.

Inheritance diagram for `clove::FlatLayout`:



Public Member Functions

- [switchInvisibleAnimationName](#) ()
Optional animation name for disabling a view.
- [setSwitchInvisibleAnimationName](#) (value)
Setter for [switchInvisibleAnimationName\(\)](#).
- [switchVisibleAnimationName](#) ()
Optional animation name for enabling a view.
- [setSwitchVisibleAnimationName](#) (value)
Setter for [switchVisibleAnimationName\(\)](#).
- [switchInvisibleAnimationDuration](#) ()
Optional animation duration (in msec) for the disabling animation.
- [setSwitchInvisibleAnimationDuration](#) (value)
Setter for [switchInvisibleAnimationDuration\(\)](#).
- [switchVisibleAnimationDuration](#) ()
Optional animation duration (in msec) for the enabling animation.
- [setSwitchVisibleAnimationDuration](#) (value)
Setter for [switchVisibleAnimationDuration\(\)](#).
- [collapseHidden](#) ()
If the non-visible widgets shall be collapsed (affects layouting).
- [setCollapseHidden](#) (value)
Setter for [collapseHidden\(\)](#).
- [currentView](#) ()
The index (integer) of the currently visible child.
- [setCurrentView](#) (value)
Setter for [currentView\(\)](#).
- [children](#) ()
List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).
- [setChildren](#) (value)
Setter for [children\(\)](#).
- [addChild](#) (value)
Adds a new child widget to the layout.
- [clearChilds](#) ()
Removes all childs from the layout.

1.43.1 Detailed Description

A mixin for flat layout implementations.

This layout shows one of its childs in full sizes and hides all the others.

1.43.2 Member Function Documentation

1.43.2.1 addChild()

```
addChild (
    value ) [inherited]
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for clove::build() .
--------------	--

1.43.2.2 children()

```
children ( ) [inherited]
```

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.43.2.3 clearChilds()

```
clearChilds ( ) [inherited]
```

Removes all childs from the layout.

1.43.2.4 collapseHidden()

```
collapseHidden ( )
```

If the non-visible widgets shall be collapsed (affects layouting).

1.43.2.5 currentView()

```
currentView ( )
```

The index (integer) of the currently visible child.

1.43.2.6 setChildren()

```
setChildren (
    value ) [inherited]
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.43.2.7 setCollapseHidden()

```
setCollapseHidden (  
    value )
```

Setter for [collapseHidden\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.43.2.8 setCurrentView()

```
setCurrentView (  
    value )
```

Setter for [currentView\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.43.2.9 setSwitchInvisibleAnimationDuration()

```
setSwitchInvisibleAnimationDuration (  
    value )
```

Setter for [switchInvisibleAnimationDuration\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.43.2.10 `setSwitchInvisibleAnimationName()`

```
setSwitchInvisibleAnimationName (
    value )
```

Setter for [switchInvisibleAnimationName\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.43.2.11 `setSwitchVisibleAnimationDuration()`

```
setSwitchVisibleAnimationDuration (
    value )
```

Setter for [switchVisibleAnimationDuration\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.43.2.12 `setSwitchVisibleAnimationName()`

```
setSwitchVisibleAnimationName (
    value )
```

Setter for [switchVisibleAnimationName\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.43.2.13 `switchInvisibleAnimationDuration()`

```
switchInvisibleAnimationDuration ( )
```

Optional animation duration (in msec) for the disabling animation.

See also [clove::FlatLayout::switchInvisibleAnimationName\(\)](#).

1.43.2.14 switchInvisibleAnimationName()

```
switchInvisibleAnimationName ( )
```

Optional animation name for disabling a view.

1.43.2.15 switchVisibleAnimationDuration()

```
switchVisibleAnimationDuration ( )
```

Optional animation duration (in msec) for the enabling animation.

See also [clove::FlatLayout::switchVisibleAnimationName\(\)](#).

1.43.2.16 switchVisibleAnimationName()

```
switchVisibleAnimationName ( )
```

Optional animation name for enabling a view.

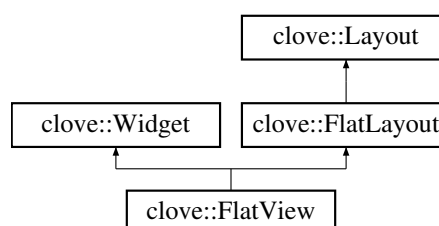
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.44 clove::FlatView Class Reference

A container which shows one of its child widgets in full size and hides all others.

Inheritance diagram for clove::FlatView:



Public Member Functions

- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)
Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)
Removes the css class `css` from this widget.
- [isStyleClass](#) (css)
Checks if this widget contains the css class `css`.
- [setStyleClassAssigned](#) (css, assigned)
Sets or unsets the css class `css` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.

- [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

- [remove](#) (removeconfig)

Removes this widget.

- [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).

- [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

- [childrenWidgets](#) ()

List of the children [clove::Widget](#) instances.

- [parentWidget](#) ()

The parent [clove::Widget](#).

- [name](#) ()

The name of the widget.

- [setName](#) (value)

Setter for [name\(\)](#).

- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).

- [setEnabled](#) (value)

Setter for [enabled\(\)](#).

- [styleClass](#) ()

Custom css class(es).

- [setStyleClass](#) (value)

Setter for [styleClass\(\)](#).

- [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

- [setStyle](#) (value)

Setter for [style\(\)](#).

- [visibility](#) ()

If this widget is visible.

- [setVisibility](#) (value)

Setter for [visibility\(\)](#).

- [doStandaloneResizing](#) ()

If the widget handles to resize itself as needed.

- [setDoStandaloneResizing](#) (value)

Setter for [doStandaloneResizing\(\)](#).

- [mayFocus](#) ()

If the widget can have the keyboard focus.

- [setMayFocus](#) (value)

Setter for [mayFocus\(\)](#).

- [horizontalStretchAffinity](#) ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

- [setHorizontalStretchAffinity](#) (value)

Setter for [horizontalStretchAffinity\(\)](#).

- [verticalStretchAffinity](#) ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

- [setVerticalStretchAffinity](#) (value)

Setter for [verticalStretchAffinity\(\)](#).

- [strictHorizontalSizing](#) ()

If the widget is strictly forbidden to get additional horizontal space.

- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.
- [switchInvisibleAnimationName](#) ()
Optional animation name for disabling a view.
- [setSwitchInvisibleAnimationName](#) (value)
Setter for [switchInvisibleAnimationName\(\)](#).
- [switchVisibleAnimationName](#) ()
Optional animation name for enabling a view.
- [setSwitchVisibleAnimationName](#) (value)
Setter for [switchVisibleAnimationName\(\)](#).
- [switchInvisibleAnimationDuration](#) ()

- Optional animation duration (in msec) for the disabling animation.*

 - [setSwitchInvisibleAnimationDuration](#) (value)

Setter for [switchInvisibleAnimationDuration\(\)](#).
- [switchVisibleAnimationDuration](#) ()

Optional animation duration (in msec) for the enabling animation.

 - [setSwitchVisibleAnimationDuration](#) (value)

Setter for [switchVisibleAnimationDuration\(\)](#).
- [collapseHidden](#) ()

If the non-visible widgets shall be collapsed (affects layouting).

 - [setCollapseHidden](#) (value)

Setter for [collapseHidden\(\)](#).
- [currentView](#) ()

The index (integer) of the currently visible child.

 - [setCurrentView](#) (value)

Setter for [currentView\(\)](#).
- [children](#) ()

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

 - [setChildren](#) (value)

Setter for [children\(\)](#).
- [addChild](#) (value)

Adds a new child widget to the layout.
- [clearChilds](#) ()

Removes all childs from the layout.

1.44.1 Detailed Description

A container which shows one of its child widgets in full size and hides all others.

1.44.2 Member Function Documentation

1.44.2.1 addChild()

```
addChild (
    value ) [inherited]
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for clove::build() .
--------------	--

1.44.2.2 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.44.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.44.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.44.2.5 children()

```
children ( ) [inherited]
```

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.44.2.6 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.44.2.7 clearChilds()

```
clearChilds ( ) [inherited]
```

Removes all childs from the layout.

1.44.2.8 collapseHidden()

```
collapseHidden ( ) [inherited]
```

If the non-visible widgets shall be collapsed (affects layouting).

1.44.2.9 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.44.2.10 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.44.2.11 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.44.2.12 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.44.2.13 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.44.2.14 currentView()

```
currentView ( ) [inherited]
```

The index (integer) of the currently visible child.

1.44.2.15 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.44.2.16 doinit()

`doinit () [inherited]`

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.44.2.17 doinitEarly()

`doinitEarly () [inherited]`

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.44.2.18 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.44.2.19 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.44.2.20 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.44.2.21 `effectiveVisibility()`

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.44.2.22 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.44.2.23 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.44.2.24 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.44.2.25 `getMinimalWidth()`

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.44.2.26 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.44.2.27 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.44.2.28 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.44.2.29 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.44.2.30 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.44.2.31 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.44.2.32 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.44.2.33 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.44.2.34 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.44.2.35 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.44.2.36 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.44.2.37 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.44.2.38 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.44.2.39 OnDestroyed()

```
OnDestroyed [inherited]
```

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.44.2.40 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.44.2.41 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.44.2.42 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.44.2.43 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.44.2.44 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.44.2.45 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.44.2.46 `outerSize()`

`outerSize () [inherited]`

Returns outer width and height of this widget.

1.44.2.47 `parentWidget()`

`parentWidget () [inherited]`

The parent [`clove::Widget`](#).

1.44.2.48 `registerBusy()`

`registerBusy () [inherited]`

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [`unregisterBusy\(\)`](#) and [`busy\(\)`](#).

You must not mix this function with direct usage of [`setBusy\(\)`](#)!

1.44.2.49 `relayout()`

`relayout () [inherited]`

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [`clove::Widget::getPreferredWidth`](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.44.2.50 `remove()`

`remove (
 removeconfig) [inherited]`

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [`clove::Widget::OnDestroyed`](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.44.2.51 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.44.2.52 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.44.2.53 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.54 setChildren()

```
setChildren (
    value ) [inherited]
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.55 setCollapseHidden()

```
setCollapseHidden (  
    value ) [inherited]
```

Setter for [collapseHidden\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.56 setCurrentView()

```
setCurrentView (  
    value ) [inherited]
```

Setter for [currentView\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.57 setDoStandaloneResizing()

```
setDoStandaloneResizing (  
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.58 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.59 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.60 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.61 `setMayFocus()`

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.62 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.63 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.44.2.64 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.65 `setStrictVerticalSizing()`

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.66 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.67 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.68 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.44.2.69 setSwitchInvisibleAnimationDuration()

```
setSwitchInvisibleAnimationDuration (  
    value ) [inherited]
```

Setter for [switchInvisibleAnimationDuration\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.70 setSwitchInvisibleAnimationName()

```
setSwitchInvisibleAnimationName (  
    value ) [inherited]
```

Setter for [switchInvisibleAnimationName\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.71 setSwitchVisibleAnimationDuration()

```
setSwitchVisibleAnimationDuration (  
    value ) [inherited]
```

Setter for [switchVisibleAnimationDuration\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.72 setSwitchVisibleAnimationName()

```
setSwitchVisibleAnimationName (
    value ) [inherited]
```

Setter for [switchVisibleAnimationName\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.73 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.74 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.75 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.44.2.76 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.44.2.77 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.44.2.78 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.44.2.79 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.44.2.80 switchInvisibleAnimationDuration()

```
switchInvisibleAnimationDuration ( ) [inherited]
```

Optional animation duration (in msec) for the disabling animation.

See also [clove::FlatLayout::switchInvisibleAnimationName\(\)](#).

1.44.2.81 switchInvisibleAnimationName()

```
switchInvisibleAnimationName ( ) [inherited]
```

Optional animation name for disabling a view.

1.44.2.82 switchVisibleAnimationDuration()

```
switchVisibleAnimationDuration ( ) [inherited]
```

Optional animation duration (in msec) for the enabling animation.

See also [clove::FlatLayout::switchVisibleAnimationName\(\)](#).

1.44.2.83 switchVisibleAnimationName()

```
switchVisibleAnimationName ( ) [inherited]
```

Optional animation name for enabling a view.

1.44.2.84 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.44.2.85 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.44.2.86 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.44.2.87 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

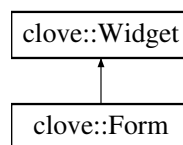
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.45 clove::Form Class Reference

A container for a form-like with a label for each entry in a row-oriented alignment.

Inheritance diagram for clove::Form:



Public Member Functions

- [sections](#) ()
A list of widget configurations as for [clove::build\(\)](#). Each defines one section in the form.
- [setSections](#) (value)
Setter for [sections\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()

- Executes late widget initialization (i.e. after properties are applied).*

 - `doinitEarly ()`
- Executes early widget initialization (i.e. before properties are applied).*

 - `resize ()`
- Applies the new widget size to internal content.*

 - `doresize ()`
- Corrects alignments of internal elements according to the new widget size.*

 - `relayout ()`
- Notifies the parent widget that a new geometry is required.*

 - `focus ()`
- Sets the focus to this widget.*

 - `isAlive ()`
- Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).*

 - `computeMinimalWidth ()`
- Computes the minimal width in pixel this widget needs to have.*

 - `computePreferredWidth ()`
- Computes the preferred width in pixel this widget needs to have.*

 - `computeMinimalHeightForWidth (w)`
- Computes the minimal height in pixel this widget needs to have for a given width.*

 - `computePreferredHeightForWidth (w)`
- Computes the preferred height in pixel this widget needs to have for a given width.*

 - `getMinimalWidth ()`
- Returns the minimal width in pixel this widget needs to have.*

 - `getPreferredWidth ()`
- Returns the preferred width in pixel this widget needs to have.*

 - `getMinimalHeightForWidth (w)`
- Returns the minimal height in pixel this widget needs to have for a given width.*

 - `getPreferredHeightForWidth (w)`
- Returns the preferred height in pixel this widget needs to have for a given width.*

 - `addStyleClass (class)`
- Adds the css class `class` to this widget.*

 - `removeStyleClass (class)`
- Removes the css class `class` from this widget.*

 - `isStyleClass (class)`
- Checks if this widget contains the css class `class`.*

 - `setStyleClassAssigned (class, assigned)`
- Sets or unsets the css class `class` to this widget.*

 - `containingNameScope ()`
- The namespace this widget contains to.*

 - `nameScope ()`
- The own namespace (or the namespace it contains to, if this widget does not bring a new one).*

 - `remove (removeconfig)`
- Removes this widget.*

 - `effectivelyEnabled ()`
- If this widget is enabled and not marked as busy (i.e. can interact with the user).*

 - `effectiveVisibility ()`
- If this widget and all parent widgets are visible. See also `visibility()`.*

 - `childrenWidgets ()`
- List of the children `clove::Widget` instances.*

 - `parentWidget ()`
- The parent `clove::Widget`.*

- `name ()`
The name of the widget.
- `setName (value)`
Setter for `name()`.
- `enabled ()`
If this widget is marked as enabled (i.e. can interact with the user).
- `setEnabled (value)`
Setter for `enabled()`.
- `styleClass ()`
Custom css class(es).
- `setStyleClass (value)`
Setter for `styleClass()`.
- `style ()`
Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle (value)`
Setter for `style()`.
- `visibility ()`
If this widget is visible.
- `setVisibility (value)`
Setter for `visibility()`.
- `doStandaloneResizing ()`
If the widget handles to resize itself as needed.
- `setDoStandaloneResizing (value)`
Setter for `doStandaloneResizing()`.
- `mayFocus ()`
If the widget can have the keyboard focus.
- `setMayFocus (value)`
Setter for `mayFocus()`.
- `horizontalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity (value)`
Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity (value)`
Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing ()`
If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing (value)`
Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing ()`
If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing (value)`
Setter for `strictVerticalSizing()`.
- `vstretch ()`
Alias for `verticalStretchAffinity()`.
- `setVstretch (value)`
Setter for `vstretch()`.
- `hstretch ()`
Alias for `horizontalStretchAffinity()`.
- `setHstretch (value)`

- Setter for `hstretch()`.
- `busy ()`
 If the widget is in busy state, typically resulting in a loading animation.
- `setBusy (value)`
 Setter for `busy()`.
- `registerBusy ()`
 Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy (token)`
 Drops a token and returns to normal state if no other tokens exist.
- `hasFocus ()`
 If the widget has the keyboard focus.
- `innerSize ()`
 Returns inner width and height of this widget.
- `outerSize ()`
 Returns outer width and height of this widget.
- `OnDestroyed`
 Triggered when this widget was removed somehow.
- `OnResized`
 Triggered when this widget was resized.
- `OnVisibilityChanged`
 Triggered when the visibility of this widget changed.
- `OnPropertyChanged`
 Triggered whenever a property of this widget changed its value.
- `OnKeyDown`
 Triggered when a keyboard key went down.
- `OnKeyUp`
 Triggered when a keyboard key came up.
- `OnKeyPress`
 Triggered when a keyboard key was pressed.

1.45.1 Detailed Description

A container for a form-like with a label for each entry in a row-oriented alignment.

1.45.2 Member Function Documentation

1.45.2.1 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.45.2.2 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.45.2.3 `busy()`

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.45.2.4 `childrenWidgets()`

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.45.2.5 `computeMinimalHeightForWidth()`

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.45.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.45.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.45.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.45.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.45.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.45.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.45.2.12 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.45.2.13 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.45.2.14 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.45.2.15 `effectivelyEnabled()`

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.45.2.16 `effectiveVisibility()`

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.45.2.17 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.45.2.18 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.45.2.19 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.45.2.20 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.45.2.21 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.45.2.22 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.45.2.23 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.45.2.24 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.45.2.25 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.45.2.26 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.45.2.27 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.45.2.28 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.45.2.29 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.45.2.30 `isStyleClass()`

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.45.2.31 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.45.2.32 `name()`

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.45.2.33 `nameScope()`

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.45.2.34 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.45.2.35 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.45.2.36 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.45.2.37 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.45.2.38 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.45.2.39 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.45.2.40 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.45.2.41 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.45.2.42 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.45.2.43 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.45.2.44 relayout()

relayout () [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.45.2.45 remove()

remove (
 removeconfig) [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.45.2.46 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.45.2.47 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.45.2.48 sections()

```
sections ( )
```

A list of widget configurations as for [clove::build\(\)](#). Each defines one section in the form.

Define the labels in the `formSectionLabel` property of each widget.

1.45.2.49 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.50 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.51 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.52 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.53 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.54 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.55 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.56 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.45.2.57 setSections()

```
setSections (
    value )
```

Setter for [sections\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.58 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.59 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.60 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.61 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.62 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.45.2.63 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.64 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.65 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.45.2.66 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.45.2.67 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.45.2.68 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.45.2.69 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.45.2.70 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.45.2.71 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.45.2.72 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.45.2.73 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

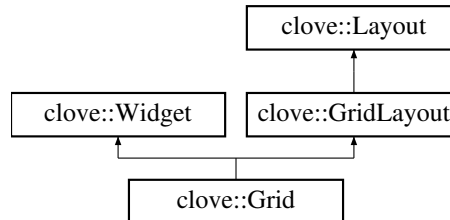
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.46 clove::Grid Class Reference

A container for aligning child widgets in a grid.

Inheritance diagram for clove::Grid:



Public Member Functions

- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.

- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addClass](#) (class)
Adds the css class `class` to this widget.
- [removeClass](#) (class)
Removes the css class `class` from this widget.
- [hasClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)

- Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
 - If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
 - Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
 - Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
 - Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
 - If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
 - Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
 - If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
 - Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
 - Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
 - Setter for [vstretch\(\)](#).
- [hstretch](#) ()
 - Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 - Setter for [hstretch\(\)](#).
- [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 - Setter for [busy\(\)](#).
- [registerBusy](#) ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
 - Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
 - If the widget has the keyboard focus.
- [innerSize](#) ()
 - Returns inner width and height of this widget.
- [outerSize](#) ()
 - Returns outer width and height of this widget.
- [OnDestroyed](#)
 - Triggered when this widget was removed somehow.
- [OnResized](#)
 - Triggered when this widget was resized.
- [OnVisibilityChanged](#)
 - Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
 - Triggered whenever a property of this widget changed its value.

- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.
- [insertRow](#) (i)
Inserts a new empty row to the grid.
- [insertColumn](#) (i)
Inserts a new empty column to the grid.
- [removeRow](#) (i)
Removes a row from the grid.
- [removeColumn](#) (i)
Removes a column from the grid.
- [children](#) ()
List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).
- [setChildren](#) (value)
Setter for [children\(\)](#).
- [addChild](#) (value)
Adds a new child widget to the layout.
- [clearChilds](#) ()
Removes all childs from the layout.

1.46.1 Detailed Description

A container for aligning child widgets in a grid.

Set the `row` and `col` property in the child widget configurations to some numbers for assigning them to cells.

1.46.2 Member Function Documentation

1.46.2.1 `addChild()`

```
addChild (
    value ) [inherited]
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for clove::build() .
--------------	--

1.46.2.2 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.46.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.46.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.46.2.5 children()

```
children ( ) [inherited]
```

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.46.2.6 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.46.2.7 clearChilds()

```
clearChilds ( ) [inherited]
```

Removes all childs from the layout.

1.46.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.46.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.46.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.46.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.46.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.46.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.46.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.46.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.46.2.16 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.46.2.17 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.46.2.18 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.46.2.19 effectiveVisibility()

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.46.2.20 enabled()

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.46.2.21 focus()

`focus () [inherited]`

Sets the focus to this widget.

1.46.2.22 getMinimalHeightForWidth()

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.46.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.46.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.46.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.46.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.46.2.27 hasFocus()

`hasFocus () [inherited]`

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.46.2.28 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.46.2.29 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.46.2.30 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.46.2.31 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.46.2.32 insertColumn()

```
insertColumn (
    i ) [inherited]
```

Inserts a new empty column to the grid.

Parameters

<i>i</i>	The column index.
----------	-------------------

1.46.2.33 insertRow()

```
insertRow (
    i ) [inherited]
```

Inserts a new empty row to the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.46.2.34 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.46.2.35 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.46.2.36 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.46.2.37 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.46.2.38 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.46.2.39 OnDestroy()

```
OnDestroyed [inherited]
```

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.46.2.40 OnKeyDown()

```
OnKeyDown [inherited]
```

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.46.2.41 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.46.2.42 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.46.2.43 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.46.2.44 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.46.2.45 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.46.2.46 `outerSize()`

```
outerSize ( ) [inherited]
```

Returns outer width and height of this widget.

1.46.2.47 `parentWidget()`

```
parentWidget ( ) [inherited]
```

The parent [Clove::Widget](#).

1.46.2.48 `registerBusy()`

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.46.2.49 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.46.2.50 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.46.2.51 removeColumn()

```
removeColumn (
    i ) [inherited]
```

Removes a column from the grid.

Parameters

<i>i</i>	The column index.
----------	-------------------

1.46.2.52 removeRow()

```
removeRow (
    i ) [inherited]
```

Removes a row from the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.46.2.53 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.46.2.54 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.46.2.55 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.56 `setChildren()`

```
setChildren (
    value ) [inherited]
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.57 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.58 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.59 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.60 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.61 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.62 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.63 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.46.2.64 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.65 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.66 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.67 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.68 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.46.2.69 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (  
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.70 `setVisibility()`

```
setVisibility (  
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.71 `setVstretch()`

```
setVstretch (  
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.46.2.72 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.46.2.73 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.46.2.74 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.46.2.75 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.46.2.76 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.46.2.77 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.46.2.78 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.46.2.79 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

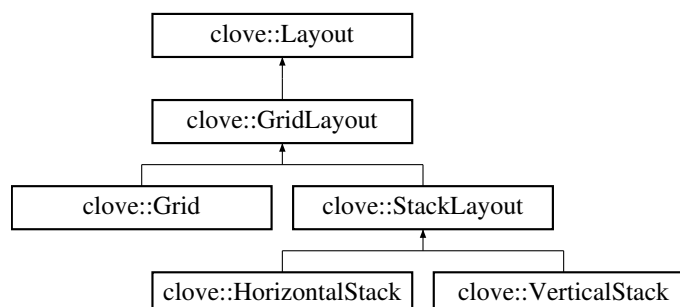
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.47 clove::GridLayout Class Reference

A mixin for grid layout implementations.

Inheritance diagram for `clove::GridLayout`:



Public Member Functions

- [insertRow](#) (i)
Inserts a new empty row to the grid.
- [insertColumn](#) (i)
Inserts a new empty column to the grid.
- [removeRow](#) (i)
Removes a row from the grid.
- [removeColumn](#) (i)
Removes a column from the grid.
- [children](#) ()
List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).
- [setChildren](#) (value)
Setter for [children\(\)](#).
- [addChild](#) (value)
Adds a new child widget to the layout.
- [clearChilds](#) ()
Removes all childs from the layout.

1.47.1 Detailed Description

A mixin for grid layout implementations.

1.47.2 Member Function Documentation

1.47.2.1 `addChild()`

```
addChild (
    value ) [inherited]
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for clove::build() .
--------------	--

1.47.2.2 `children()`

```
children ( ) [inherited]
```

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.47.2.3 clearChilds()

```
clearChilds ( ) [inherited]
```

Removes all childs from the layout.

1.47.2.4 insertColumn()

```
insertColumn (
    i )
```

Inserts a new empty column to the grid.

Parameters

<i>i</i>	The column index.
----------	-------------------

1.47.2.5 insertRow()

```
insertRow (
    i )
```

Inserts a new empty row to the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.47.2.6 removeColumn()

```
removeColumn (
    i )
```

Removes a column from the grid.

Parameters

<i>i</i>	The column index.
----------	-------------------

1.47.2.7 removeRow()

```
removeRow (
    i )
```

Removes a row from the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.47.2.8 setChildren()

```
setChildren (
    value ) [inherited]
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

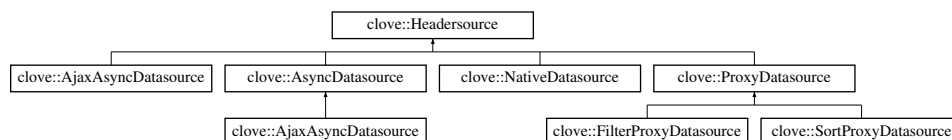
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.48 clove::Headersource Class Reference

Base class for headersources.

Inheritance diagram for clove::Headersource:



Public Member Functions

- [getRowHeader](#) (irow, parent)
Returns the header configuration for a given row.
- [getColumnHeader](#) (irow, parent)
Returns the header configuration for a given column.
- [rowHeadersVisible](#) (parent)

- If row headers are visible.*

 - [columnHeadersVisible](#) (parent)

If column headers are visible.
- [OnHeaderDataInsert](#)

Triggered when a new row or column of header data was inserted.
- [OnHeaderDataRemove](#)

Triggered when a row or column of header data was removed.
- [OnHeaderDataUpdate](#)

Triggered when header data were updated.
- [OnHeaderVisibilityUpdated](#)

Triggered when header visibilities changed.

1.48.1 Detailed Description

Base class for headersources.

A headersource provides information for headers in data views. Read the Manual for details.

1.48.2 Member Function Documentation

1.48.2.1 columnHeadersVisible()

```
columnHeadersVisible (
    parent ) [pure virtual]
```

If column headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.48.2.2 getColumnHeader()

```
getColumnHeader (
    irow,
    parent ) [pure virtual]
```

Returns the header configuration for a given column.

Parameters

<i>irow</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.48.2.3 getRowHeader()

```
getRowHeader (
    irow,
    parent ) [pure virtual]
```

Returns the header configuration for a given row.

Parameters

<i>irow</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.48.2.4 OnHeaderDataInsert()

```
OnHeaderDataInsert
```

Triggered when a new row or column of header data was inserted.

This is a [clove.Event](#) instance. See Manual for details.

1.48.2.5 OnHeaderDataRemove()

```
OnHeaderDataRemove
```

Triggered when a row or column of header data was removed.

This is a [clove.Event](#) instance. See Manual for details.

1.48.2.6 OnHeaderDataUpdate()

```
OnHeaderDataUpdate
```

Triggered when header data were updated.

This is a [clove.Event](#) instance. See Manual for details.

1.48.2.7 OnHeaderVisibilityUpdated()

```
OnHeaderVisibilityUpdated
```

Triggered when header visibilities changed.

This is a [clove.Event](#) instance. See Manual for details.

1.48.2.8 rowHeadersVisible()

```
rowHeadersVisible (
    parent ) [pure virtual]
```

If row headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

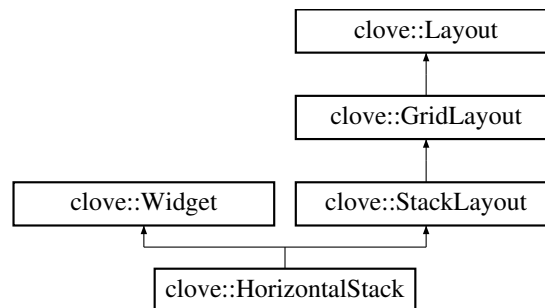
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.49 [clove::HorizontalStack](#) Class Reference

A container which stacks child widgets column-wise.

Inheritance diagram for [clove::HorizontalStack](#):



Public Member Functions

- [cols](#) ()
The list of child widgets as widget configurations like for [clove::build\(\)](#).
- [setCols](#) (value)
Setter for [cols\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.

- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (class)
Adds the css class `class` to this widget.
- [removeStyleClass](#) (class)
Removes the css class `class` from this widget.
- [isStyleClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)

- Setter for `enabled()`.
- `styleClass` ()
 - Custom css class(es).
- `setStyleClass` (value)
 - Setter for `styleClass()`.
- `style` ()
 - Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle` (value)
 - Setter for `style()`.
- `visibility` ()
 - If this widget is visible.
- `setVisibility` (value)
 - Setter for `visibility()`.
- `doStandaloneResizing` ()
 - If the widget handles to resize itself as needed.
- `setDoStandaloneResizing` (value)
 - Setter for `doStandaloneResizing()`.
- `mayFocus` ()
 - If the widget can have the keyboard focus.
- `setMayFocus` (value)
 - Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity` (value)
 - Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity` ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity` (value)
 - Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing` ()
 - If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing` (value)
 - Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing` ()
 - If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing` (value)
 - Setter for `strictVerticalSizing()`.
- `vstretch` ()
 - Alias for `verticalStretchAffinity()`.
- `setVstretch` (value)
 - Setter for `vstretch()`.
- `hstretch` ()
 - Alias for `horizontalStretchAffinity()`.
- `setHstretch` (value)
 - Setter for `hstretch()`.
- `busy` ()
 - If the widget is in busy state, typically resulting in a loading animation.
- `setBusy` (value)
 - Setter for `busy()`.
- `registerBusy` ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.

- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.
- [insertRow](#) (i)
Inserts a new empty row to the grid.
- [insertColumn](#) (i)
Inserts a new empty column to the grid.
- [removeRow](#) (i)
Removes a row from the grid.
- [removeColumn](#) (i)
Removes a column from the grid.
- [children](#) ()
List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).
- [setChildren](#) (value)
Setter for [children\(\)](#).
- [addChild](#) (value)
Adds a new child widget to the layout.
- [clearChilds](#) ()
Removes all childs from the layout.

1.49.1 Detailed Description

A container which stacks child widgets column-wise.

1.49.2 Member Function Documentation

1.49.2.1 addChild()

```
addChild (
    value ) [inherited]
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for clove::build() .
--------------	--

1.49.2.2 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.49.2.3 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.49.2.4 `busy()`

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.49.2.5 `children()`

```
children ( ) [inherited]
```

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.49.2.6 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.49.2.7 clearChilds()

```
clearChilds ( ) [inherited]
```

Removes all childs from the layout.

1.49.2.8 cols()

```
cols ( )
```

The list of child widgets as widget configurations like for [clove::build\(\)](#).

1.49.2.9 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.49.2.10 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.49.2.11 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.49.2.12 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.49.2.13 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.49.2.14 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.49.2.15 doinit()

`doinit () [inherited]`

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.49.2.16 doinitEarly()

`doinitEarly () [inherited]`

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.49.2.17 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.49.2.18 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.49.2.19 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.49.2.20 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.49.2.21 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.49.2.22 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.49.2.23 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.49.2.24 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.49.2.25 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.49.2.26 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.49.2.27 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.49.2.28 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.49.2.29 `horizontalStretchAffinity()`

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.49.2.30 `hstretch()`

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.49.2.31 `init()`

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.49.2.32 `innerSize()`

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.49.2.33 `insertColumn()`

```
insertColumn (
    i ) [inherited]
```

Inserts a new empty column to the grid.

Parameters

<i>i</i>	The column index.
----------	-------------------

1.49.2.34 insertRow()

```
insertRow (
    i ) [inherited]
```

Inserts a new empty row to the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.49.2.35 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.49.2.36 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.49.2.37 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.49.2.38 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.49.2.39 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.49.2.40 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.49.2.41 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.49.2.42 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.49.2.43 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.49.2.44 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.49.2.45 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.49.2.46 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.49.2.47 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.49.2.48 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.49.2.49 registerBusy()

`registerBusy ()` [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.49.2.50 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to `clove::Widget::getPreferredWidth` (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.49.2.51 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use `clove::Widget::OnDestroyed` for continuing after removal.

Parameters

<code>removeconfig</code>	The removal configuration (optional and only for exotic cases).
---------------------------	---

1.49.2.52 `removeColumn()`

```
removeColumn (
    i ) [inherited]
```

Removes a column from the grid.

Parameters

<code>i</code>	The column index.
----------------	-------------------

1.49.2.53 `removeRow()`

```
removeRow (
    i ) [inherited]
```

Removes a row from the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.49.2.54 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.49.2.55 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.49.2.56 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.57 setChildren()

```
setChildren (
    value ) [inherited]
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.58 setCols()

```
setCols (
    value )
```

Setter for [cols\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.59 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.60 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.61 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.62 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.63 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.64 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.65 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.49.2.66 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.67 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.68 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.69 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.70 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.49.2.71 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.72 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.73 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.49.2.74 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.49.2.75 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.49.2.76 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.49.2.77 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.49.2.78 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.49.2.79 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.49.2.80 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.49.2.81 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

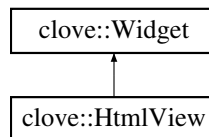
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.50 clove::HtmlView Class Reference

A host for arbitrary html content.

Inheritance diagram for clove::HtmlView:



Public Member Functions

- [contentRoot](#) ()
The root dom node of the html content.
- [setContentRoot](#) (value)
Setter for [contentRoot\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()

- Sets the focus to this widget.*

 - `isAlive ()`

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
 - `computeMinimalWidth ()`

Computes the minimal width in pixel this widget needs to have.
 - `computePreferredWidth ()`

Computes the preferred width in pixel this widget needs to have.
 - `computeMinimalHeightForWidth (w)`

Computes the minimal height in pixel this widget needs to have for a given width.
 - `computePreferredHeightForWidth (w)`

Computes the preferred height in pixel this widget needs to have for a given width.
 - `getMinimalWidth ()`

Returns the minimal width in pixel this widget needs to have.
 - `getPreferredWidth ()`

Returns the preferred width in pixel this widget needs to have.
 - `getMinimalHeightForWidth (w)`

Returns the minimal height in pixel this widget needs to have for a given width.
 - `getPreferredHeightForWidth (w)`

Returns the preferred height in pixel this widget needs to have for a given width.
 - `addStyleClass (class)`

Adds the css class `class` to this widget.
 - `removeStyleClass (class)`

Removes the css class `class` from this widget.
 - `isStyleClass (class)`

Checks if this widget contains the css class `class`.
 - `setStyleClassAssigned (class, assigned)`

Sets or unsets the css class `class` to this widget.
 - `containingNameScope ()`

The namespace this widget contains to.
 - `nameScope ()`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
 - `remove (removeconfig)`

Removes this widget.
 - `effectivelyEnabled ()`

If this widget is enabled and not marked as busy (i.e. can interact with the user).
 - `effectiveVisibility ()`

If this widget and all parent widgets are visible. See also `visibility()`.
 - `childrenWidgets ()`

List of the children `clove::Widget` instances.
 - `parentWidget ()`

The parent `clove::Widget`.
 - `name ()`

The name of the widget.
 - `setName (value)`

Setter for `name()`.
 - `enabled ()`

If this widget is marked as enabled (i.e. can interact with the user).
 - `setEnabled (value)`

Setter for `enabled()`.
 - `styleClass ()`

Custom css class(es).

- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()

If the widget has the keyboard focus.

- [innerSize \(\)](#)
Returns inner width and height of this widget.
- [outerSize \(\)](#)
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.50.1 Detailed Description

A host for arbitrary html content.

1.50.2 Member Function Documentation

1.50.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.50.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.50.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.50.2.4 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.50.2.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.50.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.50.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.50.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.50.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.50.2.10 contentRoot()

```
contentRoot ( )
```

The root dom node of the html content.

1.50.2.11 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.50.2.12 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.50.2.13 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.50.2.14 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.50.2.15 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.50.2.16 `effectivelyEnabled()`

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.50.2.17 `effectiveVisibility()`

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.50.2.18 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.50.2.19 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.50.2.20 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.50.2.21 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.50.2.22 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.50.2.23 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.50.2.24 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.50.2.25 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.50.2.26 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.50.2.27 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.50.2.28 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.50.2.29 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.50.2.30 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.50.2.31 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.50.2.32 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.50.2.33 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.50.2.34 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.50.2.35 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.50.2.36 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.50.2.37 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.50.2.38 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.50.2.39 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.50.2.40 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.50.2.41 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.50.2.42 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.50.2.43 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.50.2.44 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.50.2.45 relayout()

relayout () [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.50.2.46 remove()

remove (
 removeconfig) [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.50.2.47 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.50.2.48 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.50.2.49 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.50 `setContentRoot()`

```
setContentRoot (
    value )
```

Setter for [contentRoot\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.51 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.52 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.53 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.54 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.55 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.56 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.57 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.50.2.58 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.59 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.60 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.61 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.62 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.50.2.63 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.64 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.65 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.50.2.66 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.50.2.67 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.50.2.68 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.50.2.69 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.50.2.70 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.50.2.71 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.50.2.72 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.50.2.73 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.51 clove::l18N Class Reference

Internationalization support class.

Public Member Functions

- [addString](#) (id, texts)
Adds a text to the internationalization text table.
- [setLanguage](#) (langcode)
Sets the current language.
- static [parseLangCode](#) (lang)
Returns a clove string representation.

1.51.1 Detailed Description

Internationalization support class.

Always use the instance [clove.i18n](#). Read the Manual for details.

1.51.2 Member Function Documentation

1.51.2.1 addString()

```
addString (
    id,
    texts )
```

Adds a text to the internationalization text table.

The keys are ISO 639-1 language codes.

Example: `'addString('PleaseWait', {'de':'Bitte warten', 'en':'Please wait'})'`

Parameters

<i>id</i>	The text identifier name, like 'FooBar'.
<i>texts</i>	A configuration object which holds a text representation for all target languages.

1.51.2.2 parseLangCode()

```
static parseLangCode (
    lang )
```

Returns a clove string representation.

Only intended to be used by the infrastructure.

Parameters

<i>lang</i>	A language code string as understood by addString() .
-------------	---

1.51.2.3 setLanguage()

```
setLanguage (
    langcode )
```

Sets the current language.

You don't need to call this function, unless you want to override the user's language as it is configured in the operating system or browser.

Parameters

<i>langcode</i>	The new current language (as ISO 639-1 language code).
-----------------	--

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.52 clove::Icon Class Reference

An icon.

Public Member Functions

- static [byUrl](#) (url)
Creates a [clove::Icon](#) by a url pointing to an image.
- static [bySymbol](#) (chr)
Creates a [clove::Icon](#) by a text character.
- [createHtml](#) (height)
Creates an html representation.

1.52.1 Detailed Description

An icon.

Can be shown at many places, like in buttons, menus or [clove::IconView](#).

Construct instances with [clove::Icon::byUrl\(\)](#) or [clove::Icon::bySymbol\(\)](#).

1.52.2 Member Function Documentation

1.52.2.1 bySymbol()

```
static bySymbol (
    chr )
```

Creates a [clove::Icon](#) by a text character.

Parameters

<i>chr</i>	The text character.
------------	---------------------

1.52.2.2 byUrl()

```
static byUrl (
    url )
```

Creates a [clove::Icon](#) by a url pointing to an image.

Parameters

<i>url</i>	The image url.
------------	----------------

1.52.2.3 createHtml()

```
createHtml (
    height )
```

Creates an html representation.

This method is used by widget implementations.

Parameters

<i>height</i>	The icon height in pixels.
---------------	----------------------------

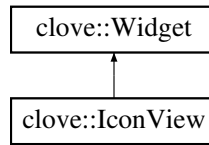
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.53 clove::IconView Class Reference

Shows an icon.

Inheritance diagram for clove::IconView:



Public Member Functions

- [size](#) ()
The icon size as css length.
- [setSize](#) (value)
Setter for [size\(\)](#).
- [icon](#) ()
The [clove::Icon](#) to show.
- [setIcon](#) (value)
Setter for [icon\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.

- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)
Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)
Removes the css class `css` from this widget.
- [isStyleClass](#) (css)
Checks if this widget contains the css class `css`.
- [setStyleClassAssigned](#) (css, assigned)
Sets or unsets the css class `css` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()

- If this widget is visible.*

 - `setVisibility` (value)
 Setter for `visibility()`.
- `doStandaloneResizing` ()
If the widget handles to resize itself as needed.

 - `setDoStandaloneResizing` (value)
 Setter for `doStandaloneResizing()`.
- `mayFocus` ()
If the widget can have the keyboard focus.

 - `setMayFocus` (value)
 Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

 - `setHorizontalStretchAffinity` (value)
 Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

 - `setVerticalStretchAffinity` (value)
 Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing` ()
If the widget is strictly forbidden to get additional horizontal space.

 - `setStrictHorizontalSizing` (value)
 Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing` ()
If the widget is strictly forbidden to get additional vertical space.

 - `setStrictVerticalSizing` (value)
 Setter for `strictVerticalSizing()`.
- `vstretch` ()
Alias for `verticalStretchAffinity()`.

 - `setVstretch` (value)
 Setter for `vstretch()`.
- `hstretch` ()
Alias for `horizontalStretchAffinity()`.

 - `setHstretch` (value)
 Setter for `hstretch()`.
- `busy` ()
If the widget is in busy state, typically resulting in a loading animation.

 - `setBusy` (value)
 Setter for `busy()`.
- `registerBusy` ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy` (token)
Drops a token and returns to normal state if no other tokens exist.
- `hasFocus` ()
If the widget has the keyboard focus.
- `innerSize` ()
Returns inner width and height of this widget.
- `outerSize` ()
Returns outer width and height of this widget.
- `OnDestroyed`
Triggered when this widget was removed somehow.

- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.53.1 Detailed Description

Shows an icon.

The icon can come from an image file or from a Unicode symbol.

1.53.2 Member Function Documentation

1.53.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.53.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<code>k</code>	The widget property name.
<code>vb</code>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.53.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.53.2.4 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.53.2.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.53.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.53.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.53.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.53.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.53.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.53.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.53.2.12 `doinitEarly()`

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.53.2.13 `doresize()`

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.53.2.14 `doStandaloneResizing()`

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.53.2.15 `effectivelyEnabled()`

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.53.2.16 `effectiveVisibility()`

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.53.2.17 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.53.2.18 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.53.2.19 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.53.2.20 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.53.2.21 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.53.2.22 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.53.2.23 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.53.2.24 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.53.2.25 `horizontalStretchAffinity()`

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.53.2.26 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.53.2.27 icon()

`icon ()`

The [clove::Icon](#) to show.

1.53.2.28 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.53.2.29 innerSize()

`innerSize () [inherited]`

Returns inner width and height of this widget.

1.53.2.30 isAlive()

`isAlive () [inherited]`

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.53.2.31 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.53.2.32 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.53.2.33 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.53.2.34 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.53.2.35 OnDestroyed()

```
OnDestroyed [inherited]
```

Triggered when this widget was removed somehow.

This is a [Clove.Event](#) instance. See Manual for details.

1.53.2.36 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.53.2.37 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.53.2.38 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.53.2.39 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.53.2.40 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.53.2.41 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.53.2.42 `outerSize()`

```
outerSize ( ) [inherited]
```

Returns outer width and height of this widget.

1.53.2.43 `parentWidget()`

```
parentWidget ( ) [inherited]
```

The parent [Clove::Widget](#).

1.53.2.44 `registerBusy()`

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.53.2.45 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.53.2.46 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.53.2.47 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.53.2.48 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.53.2.49 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.50 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.51 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.52 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.53 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.54 setIcon()

```
setIcon (
    value )
```

Setter for [icon\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.55 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.56 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.57 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.53.2.58 setSize()

```
setSize (
    value )
```

Setter for [size\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.59 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.60 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.61 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.62 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.63 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.53.2.64 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.65 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.66 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.53.2.67 size()

```
size ( )
```

The icon size as css length.

1.53.2.68 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretch←Affinity\(\)](#)==0.

1.53.2.69 strictVerticalSizing()

`strictVerticalSizing () [inherited]`

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.53.2.70 style()

`style () [inherited]`

Custom css style string. You should not use that, but `styleClass()` instead.

1.53.2.71 styleClass()

`styleClass () [inherited]`

Custom css class(es).

1.53.2.72 unregisterBusy()

`unregisterBusy (
 token) [inherited]`

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.53.2.73 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.53.2.74 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.53.2.75 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

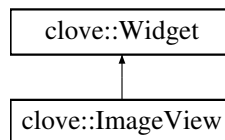
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.54 clove::ImageView Class Reference

Shows an image.

Inheritance diagram for `clove::ImageView`:



Public Member Functions

- [source](#) ()
The image source URL.
- [setSource](#) (value)
Setter for [source\(\)](#).
- [zoom](#) ()
The zoom level.
- [setZoom](#) (value)
Setter for [zoom\(\)](#).
- [keepAspectRatio](#) ()
For 'auto' zoom: Whether to keep aspect ratio of the image.
- [setKeepAspectRatio](#) (value)
Setter for [keepAspectRatio\(\)](#).
- [OnLoaded](#)
Triggered when the image loading ended.
- [declareProperty](#) (k, defaultV)

- Declares a widget property.*
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read [Manual](#) for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)
Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)
Removes the css class `css` from this widget.
- [isStyleClass](#) (css)
Checks if this widget contains the css class `css`.
- [setStyleClassAssigned](#) (css, assigned)
Sets or unsets the css class `css` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).

- **remove** (removeconfig)
Removes this widget.
- **effectivelyEnabled** ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- **effectiveVisibility** ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- **childrenWidgets** ()
List of the children [clove::Widget](#) instances.
- **parentWidget** ()
The parent [clove::Widget](#).
- **name** ()
The name of the widget.
- **setName** (value)
Setter for [name\(\)](#).
- **enabled** ()
If this widget is marked as enabled (i.e. can interact with the user).
- **setEnabled** (value)
Setter for [enabled\(\)](#).
- **styleClass** ()
Custom css class(es).
- **setStyleClass** (value)
Setter for [styleClass\(\)](#).
- **style** ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- **setStyle** (value)
Setter for [style\(\)](#).
- **visibility** ()
If this widget is visible.
- **setVisibility** (value)
Setter for [visibility\(\)](#).
- **doStandaloneResizing** ()
If the widget handles to resize itself as needed.
- **setDoStandaloneResizing** (value)
Setter for [doStandaloneResizing\(\)](#).
- **mayFocus** ()
If the widget can have the keyboard focus.
- **setMayFocus** (value)
Setter for [mayFocus\(\)](#).
- **horizontalStretchAffinity** ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- **setHorizontalStretchAffinity** (value)
Setter for [horizontalStretchAffinity\(\)](#).
- **verticalStretchAffinity** ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- **setVerticalStretchAffinity** (value)
Setter for [verticalStretchAffinity\(\)](#).
- **strictHorizontalSizing** ()
If the widget is strictly forbidden to get additional horizontal space.
- **setStrictHorizontalSizing** (value)
Setter for [strictHorizontalSizing\(\)](#).
- **strictVerticalSizing** ()

- If the widget is strictly forbidden to get additional vertical space.*
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.54.1 Detailed Description

Shows an image.

This does not contain any viewer controls like zoom buttons.

1.54.2 Member Function Documentation

1.54.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.54.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<code>k</code>	The widget property name.
<code>vb</code>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.54.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.54.2.4 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.54.2.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.54.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.54.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.54.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.54.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.54.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.54.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.54.2.12 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.54.2.13 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.54.2.14 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.54.2.15 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.54.2.16 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.54.2.17 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.54.2.18 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.54.2.19 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.54.2.20 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.54.2.21 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.54.2.22 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.54.2.23 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.54.2.24 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.54.2.25 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.54.2.26 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.54.2.27 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.54.2.28 `innerSize()`

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.54.2.29 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.54.2.30 `isStyleClass()`

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.54.2.31 `keepAspectRatio()`

```
keepAspectRatio ( )
```

For 'auto' zoom: Whether to keep aspect ratio of the image.

1.54.2.32 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.54.2.33 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.54.2.34 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.54.2.35 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.54.2.36 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.54.2.37 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.54.2.38 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.54.2.39 OnLoaded()

`OnLoaded`

Triggered when the image loading ended.

This is a [clove.Event](#) instance. See Manual for details.

1.54.2.40 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.54.2.41 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.54.2.42 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.54.2.43 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.54.2.44 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.54.2.45 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.54.2.46 relayayout()

```
relayayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.54.2.47 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.54.2.48 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.54.2.49 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.54.2.50 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.51 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.52 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.53 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (  
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.54 setHstretch()

```
setHstretch (  
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.55 setKeepAspectRatio()

```
setKeepAspectRatio (  
    value )
```

Setter for [keepAspectRatio\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.56 setMayFocus()

```
setMayFocus (  
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.57 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.58 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.54.2.59 setSource()

```
setSource (
    value )
```

Setter for [source\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.60 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.61 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.62 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.63 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.64 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.54.2.65 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.66 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.67 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.68 setZoom()

```
setZoom (
    value )
```

Setter for [zoom\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.54.2.69 source()

```
source ( )
```

The image source URL.

1.54.2.70 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.54.2.71 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.54.2.72 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.54.2.73 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.54.2.74 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.54.2.75 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.54.2.76 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.54.2.77 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

1.54.2.78 zoom()

```
zoom ( )
```

The zoom level.

1.0 is normal, larger values zoom in, smaller values zoom out. Specify 'auto' for automatically adjusting it to the available room.

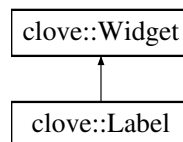
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.55 clove::Label Class Reference

A text label.

Inheritance diagram for clove::Label:



Public Member Functions

- [label](#) ()
The label text.
- [setLabel](#) (value)
Setter for [label\(\)](#).
- [htmlContent](#) ()
The label content as html.
- [setHtmlContent](#) (value)
Setter for [htmlContent\(\)](#).
- [focusBuddy](#) ()
A [clove::Widget](#) or a widget name for a focus buddy.
- [setFocusBuddy](#) (value)
Setter for [focusBuddy\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.

- `getProperty` (k)
General-purpose getter for widget properties.
- `setProperty` (k, v)
General-purpose setter for widget properties.
- `bindProperty` (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- `init` (rootNameScope)
Initializes the widget.
- `doinit` ()
Executes late widget initialization (i.e. after properties are applied).
- `doinitEarly` ()
Executes early widget initialization (i.e. before properties are applied).
- `resize` ()
Applies the new widget size to internal content.
- `doresize` ()
Corrects alignments of internal elements according to the new widget size.
- `relayout` ()
Notifies the parent widget that a new geometry is required.
- `focus` ()
Sets the focus to this widget.
- `isAlive` ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- `computeMinimalWidth` ()
Computes the minimal width in pixel this widget needs to have.
- `computePreferredWidth` ()
Computes the preferred width in pixel this widget needs to have.
- `computeMinimalHeightForWidth` (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- `computePreferredHeightForWidth` (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- `getMinimalWidth` ()
Returns the minimal width in pixel this widget needs to have.
- `getPreferredWidth` ()
Returns the preferred width in pixel this widget needs to have.
- `getMinimalHeightForWidth` (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- `getPreferredHeightForWidth` (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- `addStyleClass` (css)
Adds the css class `css` to this widget.
- `removeStyleClass` (css)
Removes the css class `css` from this widget.
- `isStyleClass` (css)
Checks if this widget contains the css class `css`.
- `setStyleClassAssigned` (css, assigned)
Sets or unsets the css class `css` to this widget.
- `containingNameScope` ()
The namespace this widget contains to.
- `nameScope` ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- `remove` (removeconfig)

- Removes this widget.*

 - [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
 - [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
 - [childrenWidgets](#) ()

List of the children [clove::Widget](#) instances.
 - [parentWidget](#) ()

The parent [clove::Widget](#).
 - [name](#) ()

The name of the widget.
 - [setName](#) (value)

Setter for [name\(\)](#).
 - [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
 - [setEnabled](#) (value)

Setter for [enabled\(\)](#).
 - [styleClass](#) ()

Custom css class(es).
 - [setStyleClass](#) (value)

Setter for [styleClass\(\)](#).
 - [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
 - [setStyle](#) (value)

Setter for [style\(\)](#).
 - [visibility](#) ()

If this widget is visible.
 - [setVisibility](#) (value)

Setter for [visibility\(\)](#).
 - [doStandaloneResizing](#) ()

If the widget handles to resize itself as needed.
 - [setDoStandaloneResizing](#) (value)

Setter for [doStandaloneResizing\(\)](#).
 - [mayFocus](#) ()

If the widget can have the keyboard focus.
 - [setMayFocus](#) (value)

Setter for [mayFocus\(\)](#).
 - [horizontalStretchAffinity](#) ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
 - [setHorizontalStretchAffinity](#) (value)

Setter for [horizontalStretchAffinity\(\)](#).
 - [verticalStretchAffinity](#) ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
 - [setVerticalStretchAffinity](#) (value)

Setter for [verticalStretchAffinity\(\)](#).
 - [strictHorizontalSizing](#) ()

If the widget is strictly forbidden to get additional horizontal space.
 - [setStrictHorizontalSizing](#) (value)

Setter for [strictHorizontalSizing\(\)](#).
 - [strictVerticalSizing](#) ()

If the widget is strictly forbidden to get additional vertical space.

- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.55.1 Detailed Description

A text label.

1.55.2 Member Function Documentation

1.55.2.1 [addStyleClass\(\)](#)

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.55.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.55.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.55.2.4 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.55.2.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.55.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.55.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.55.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.55.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.55.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.55.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.55.2.12 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.55.2.13 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.55.2.14 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.55.2.15 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.55.2.16 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.55.2.17 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.55.2.18 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.55.2.19 focusBuddy()

```
focusBuddy ( )
```

A [Clove::Widget](#) or a widget name for a focus buddy.

Whenever the user clicks on this label, it will focus the buddy.

1.55.2.20 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.55.2.21 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.55.2.22 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.55.2.23 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.55.2.24 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.55.2.25 `hasFocus()`

`hasFocus () [inherited]`

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.55.2.26 `horizontalStretchAffinity()`

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.55.2.27 `hstretch()`

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.55.2.28 `htmlContent()`

`htmlContent ()`

The label content as html.

For plain text, use [label\(\)](#) instead.

1.55.2.29 `init()`

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.55.2.30 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.55.2.31 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.55.2.32 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.55.2.33 label()

```
label ( )
```

The label text.

For rich content, use [htmlContent\(\)](#) instead.

1.55.2.34 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.55.2.35 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.55.2.36 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.55.2.37 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.55.2.38 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.55.2.39 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.55.2.40 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.55.2.41 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.55.2.42 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.55.2.43 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.55.2.44 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.55.2.45 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.55.2.46 registerBusy()

`registerBusy ()` [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.55.2.47 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.55.2.48 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.55.2.49 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.55.2.50 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.55.2.51 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.52 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.53 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.54 setFocusBuddy()

```
setFocusBuddy (
    value )
```

Setter for [focusBuddy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.55 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (  
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.56 setHstretch()

```
setHstretch (  
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.57 setHtmlContent()

```
setHtmlContent (  
    value )
```

Setter for [htmlContent\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.58 setLabel()

```
setLabel (
    value )
```

Setter for [label\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.59 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.60 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.61 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.55.2.62 `setStrictHorizontalSizing()`

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.63 `setStrictVerticalSizing()`

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.64 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.65 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.66 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.55.2.67 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.68 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.69 `setVstretch()`

```
setVstretch (
    value ) [inherited]
```

Setter for `vstretch()`.

Parameters

<i>value</i>	The new value.
--------------	----------------

1.55.2.70 `strictHorizontalSizing()`

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.55.2.71 `strictVerticalSizing()`

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.55.2.72 `style()`

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.55.2.73 styleClass()

styleClass () [inherited]

Custom css class(es).

1.55.2.74 unregisterBusy()

unregisterBusy (
 token) [inherited]

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.55.2.75 verticalStretchAffinity()

verticalStretchAffinity () [inherited]

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.55.2.76 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.55.2.77 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

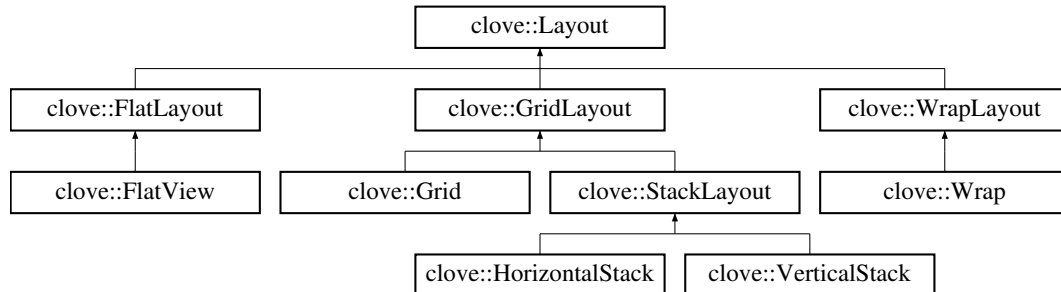
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.56 clove::Layout Class Reference

A mixin for a widget classes which align child widgets in some way.

Inheritance diagram for clove::Layout:



Public Member Functions

- [children](#) ()
List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).
- [setChildren](#) (value)
Setter for [children\(\)](#).
- [addChild](#) (value)
Adds a new child widget to the layout.
- [clearChilds](#) ()
Removes all childs from the layout.

1.56.1 Detailed Description

A mixin for a widget classes which align child widgets in some way.

This is a base type and has some more interesting sub-classes.

1.56.2 Member Function Documentation

1.56.2.1 addChild()

```
addChild (
    value )
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for clove::build() .
--------------	--

1.56.2.2 children()

```
children ( )
```

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.56.2.3 clearChilds()

```
clearChilds ( )
```

Removes all childs from the layout.

1.56.2.4 setChildren()

```
setChildren (
    value )
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

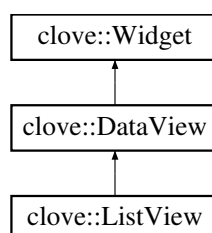
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.57 clove::ListView Class Reference

A view which shows a [clove::Datasource](#) in a list.

Inheritance diagram for clove::ListView:



Public Member Functions

- [datasource](#) ()
The [Clove::Datasource](#) for this view.
- [setDatasource](#) (value)
Setter for [datasource\(\)](#).
- [dataViewProcessor](#) ()
An optional function (ptr, value) for processing a datasource value to a display string.
- [setDataViewProcessor](#) (value)
Setter for [dataViewProcessor\(\)](#).
- [cellGenerator](#) ()
A generator function for complex cell content.
- [setCellGenerator](#) (value)
Setter for [cellGenerator\(\)](#).
- [alwaysAllocateExpanderSpace](#) ()
If some space is allocated for an expander even for cells without children.
- [setAlwaysAllocateExpanderSpace](#) (value)
Setter for [alwaysAllocateExpanderSpace\(\)](#).
- [showOnlyFirstColumn](#) ()
If to show only the first column and hide the other ones.
- [setShowOnlyFirstColumn](#) (value)
Setter for [showOnlyFirstColumn\(\)](#).
- [hideExpanders](#) ()
If to hide all expanders.
- [setHideExpanders](#) (value)
Setter for [hideExpanders\(\)](#).
- [allowSelection](#) ()
If it is allowed to select nodes.
- [setAllowSelection](#) (value)
Setter for [allowSelection\(\)](#).
- [allowChecking](#) ()
If it is allowed to check cells.
- [setAllowChecking](#) (value)
Setter for [allowChecking\(\)](#).
- [granularity](#) ()
The granularity for stuff like selection and checking.
- [setGranularity](#) (value)
Setter for [granularity\(\)](#).
- [showChangeMenu](#) ()
If a menu shall be shown for making manual changes to the data source.
- [setShowChangeMenu](#) (value)
Setter for [showChangeMenu\(\)](#).
- [editOnGesture](#) ()
If the user shall be able to edit cells by double clicking/tapping them.
- [setEditOnGesture](#) (value)
Setter for [editOnGesture\(\)](#).
- [rowsResizable](#) ()
If the user shall be able to resize rows.
- [setRowsResizable](#) (value)
Setter for [rowsResizable\(\)](#).
- [columnsResizable](#) ()

- If the user shall be able to resize columns.*

 - [setColumnsResizable](#) (value)
Setter for [columnsResizable\(\)](#).
- [selectCell](#) (ptr)
Selects a cell.
- [isCellSelected](#) (ptr)
Checks if a given cell is selected.
- [checkedCells](#) ()
Returns the checked cells as list of [clove::DatasourceValuePointer](#).
- [setCheckedCells](#) (ptrs)
Returns the checked cells.
- [isCellChecked](#) (ptr)
Checks if a given cell is checked.
- [setCellChecked](#) (ptr, val)
Sets if a given cell is checked.
- [selection](#) ()
Returns the selected item as [clove::DatasourceValuePointer](#).
- [headersource](#) ()
The [clove::Headersource](#) providing row and column header configurations.
- [setHeadersource](#) (value)
Setter for [headersource\(\)](#).
- [gridVisible](#) ()
If to show a cell grid.
- [setGridVisible](#) (value)
Setter for [gridVisible\(\)](#).
- [nodeActivationNeedsDoubleClick](#) ()
If node activation needs a double-click.
- [setNodeActivationNeedsDoubleClick](#) (value)
Setter for [nodeActivationNeedsDoubleClick\(\)](#).
- [expandCell](#) (ptr)
Expands a given cell.
- [expandCellRecursive](#) (ptr)
Expands a given cell and all parents.
- [collapseCell](#) (ptr)
Collapses a given cell.
- [isCellExpanded](#) (ptr)
Checks if a given cell is expanded.
- [editCell](#) (ptr)
Triggers the edit mode for a cell.
- [OnSelectionChanged](#)
Triggered when the selection in the view changes.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.

- [doinit \(\)](#)
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly \(\)](#)
Executes early widget initialization (i.e. before properties are applied).
- [resize \(\)](#)
Applies the new widget size to internal content.
- [doresize \(\)](#)
Corrects alignments of internal elements according to the new widget size.
- [relayout \(\)](#)
Notifies the parent widget that a new geometry is required.
- [focus \(\)](#)
Sets the focus to this widget.
- [isAlive \(\)](#)
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth \(\)](#)
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth \(\)](#)
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth \(w\)](#)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth \(w\)](#)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth \(\)](#)
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth \(\)](#)
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth \(w\)](#)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth \(w\)](#)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass \(class\)](#)
Adds the css class `class` to this widget.
- [removeStyleClass \(class\)](#)
Removes the css class `class` from this widget.
- [isStyleClass \(class\)](#)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned \(class, assigned\)](#)
Sets or unsets the css class `class` to this widget.
- [containingNameScope \(\)](#)
The namespace this widget contains to.
- [nameScope \(\)](#)
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove \(removeconfig\)](#)
Removes this widget.
- [effectivelyEnabled \(\)](#)
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility \(\)](#)
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets \(\)](#)
List of the children `clove::Widget` instances.
- [parentWidget \(\)](#)

The parent [clove::Widget](#).

- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).

- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.57.1 Detailed Description

A view which shows a [clove::Datasource](#) in a list.

1.57.2 Member Function Documentation

1.57.2.1 [addStyleClass\(\)](#)

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.57.2.2 allowChecking()

```
allowChecking ( ) [inherited]
```

If it is allowed to check cells.

This is a way to select 0..n data cells. For a single-selection, please check [allowSelection\(\)](#).

1.57.2.3 allowSelection()

```
allowSelection ( ) [inherited]
```

If it is allowed to select nodes.

This is always a single selection. For something like multi-selection, please check [allowChecking\(\)](#).

1.57.2.4 alwaysAllocateExpanderSpace()

```
alwaysAllocateExpanderSpace ( ) [inherited]
```

If some space is allocated for an expander even for cells without children.

1.57.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.57.2.6 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.57.2.7 cellGenerator()

```
cellGenerator ( ) [inherited]
```

A generator function for complex cell content.

This method is called for each cell with ([clove::Widget](#), [clove::DatasourceValuePointer](#)). It may return a widget configuration as for [clove::build\(\)](#). If it does, the cell is constructed with this widget as content. The content must define an `update(clove::Widget, clove::DatasourceValuePointer, value)` method, which takes the cell datasource value and configures the inner widget according to that.

1.57.2.8 checkedCells()

```
checkedCells ( ) [inherited]
```

Returns the checked cells as list of [clove::DatasourceValuePointer](#).

1.57.2.9 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.57.2.10 collapseCell()

```
collapseCell (
    ptr ) [inherited]
```

Collapses a given cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.57.2.11 columnsResizable()

```
columnsResizable ( ) [inherited]
```

If the user shall be able to resize columns.

1.57.2.12 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.57.2.13 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.57.2.14 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.57.2.15 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.57.2.16 `containingNameScope()`

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.57.2.17 `datasource()`

```
datasource ( ) [inherited]
```

The [clove::Datasource](#) for this view.

This provides the actual data to display. See [clove::NativeDatasource](#) for a ready-to-use implementation.

1.57.2.18 `dataViewProcessor()`

```
dataViewProcessor ( ) [inherited]
```

An optional `function(ptr, value)` for processing a `datasource` value to a display string.

1.57.2.19 `declareProperty()`

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.57.2.20 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.57.2.21 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.57.2.22 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.57.2.23 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.57.2.24 editCell()

```
editCell (
    ptr ) [inherited]
```

Triggers the edit mode for a cell.

Only works if a method `_getdomcell(ir, ic, parent)` returns a dom node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.57.2.25 editOnGesture()

```
editOnGesture ( ) [inherited]
```

If the user shall be able to edit cells by double clicking/tapping them.

1.57.2.26 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.57.2.27 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.57.2.28 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.57.2.29 expandCell()

```
expandCell (
    ptr ) [inherited]
```

Expands a given cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.57.2.30 expandCellRecursive()

```
expandCellRecursive (
    ptr ) [inherited]
```

Expands a given cell and all parents.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.57.2.31 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.57.2.32 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.57.2.33 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.57.2.34 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.57.2.35 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.57.2.36 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.57.2.37 `granularity()`

```
granularity ( ) [inherited]
```

The granularity for stuff like selection and checking.

Either `'cell'` or `'row'`.

1.57.2.38 `gridVisible()`

`gridVisible ()` [inherited]

If to show a cell grid.

1.57.2.39 `hasFocus()`

`hasFocus ()` [inherited]

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.57.2.40 `headersource()`

`headersource ()` [inherited]

The [clove::Headersource](#) providing row and column header configurations.

1.57.2.41 `hideExpanders()`

`hideExpanders ()` [inherited]

If to hide all expanders.

1.57.2.42 `horizontalStretchAffinity()`

`horizontalStretchAffinity ()` [inherited]

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.57.2.43 `hstretch()`

`hstretch ()` [inherited]

Alias for [horizontalStretchAffinity\(\)](#).

1.57.2.44 `init()`

`init (
 rootNameScope)` [inherited]

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.57.2.45 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.57.2.46 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.57.2.47 isCellChecked()

```
isCellChecked (
    ptr ) [inherited]
```

Checks if a given cell is checked.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.57.2.48 isCellExpanded()

```
isCellExpanded (
    ptr ) [inherited]
```

Checks if a given cell is expanded.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.57.2.49 isCellSelected()

```
isCellSelected (
    ptr ) [inherited]
```

Checks if a given cell is selected.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.57.2.50 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.57.2.51 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.57.2.52 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.57.2.53 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.57.2.54 nodeActivationNeedsDoubleClick()

`nodeActivationNeedsDoubleClick () [inherited]`

If node activation needs a double-click.

Note: If you set this, you should find alternative ways for users of mobile devices!

1.57.2.55 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.57.2.56 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.57.2.57 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.57.2.58 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.57.2.59 OnPropertyChanged()

`OnPropertyChanged [inherited]`

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.57.2.60 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.57.2.61 OnSelectionChanged()

OnSelectionChanged [inherited]

Triggered when the selection in the view changes.

This is a [clove.Event](#) instance. See Manual for details.

1.57.2.62 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.57.2.63 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.57.2.64 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.57.2.65 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.57.2.66 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.57.2.67 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.57.2.68 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.57.2.69 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.57.2.70 rowsResizable()

```
rowsResizable ( ) [inherited]
```

If the user shall be able to resize rows.

1.57.2.71 selectCell()

```
selectCell (
    ptr ) [inherited]
```

Selects a cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.57.2.72 selection()

```
selection ( ) [pure virtual], [inherited]
```

Returns the selected item as [clove::DatasourceValuePointer](#).

1.57.2.73 setAllowChecking()

```
setAllowChecking (
    value ) [inherited]
```

Setter for [allowChecking\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.74 setAllowSelection()

```
setAllowSelection (
    value ) [inherited]
```

Setter for [allowSelection\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.75 setAlwaysAllocateExpanderSpace()

```
setAlwaysAllocateExpanderSpace (  
    value ) [inherited]
```

Setter for [alwaysAllocateExpanderSpace\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.76 setBusy()

```
setBusy (  
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.77 setCellChecked()

```
setCellChecked (  
    ptr,  
    val ) [inherited]
```

Sets if a given cell is checked.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>val</i>	If the cells shall be checked.

1.57.2.78 setCellGenerator()

```
setCellGenerator (
    value ) [inherited]
```

Setter for [cellGenerator\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.79 setCheckedCells()

```
setCheckedCells (
    ptrs ) [inherited]
```

Returns the checked cells.

Parameters

<i>ptrs</i>	A list of clove::DatasourceValuePointer .
-------------	---

1.57.2.80 setColumnsResizable()

```
setColumnsResizable (
    value ) [inherited]
```

Setter for [columnsResizable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.81 setDatasource()

```
setDatasource (
    value ) [inherited]
```

Setter for [datasource\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.82 setDataViewProcessor()

```
setDataViewProcessor (
    value ) [inherited]
```

Setter for [dataViewProcessor\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.83 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.84 setEditOnGesture()

```
setEditOnGesture (
    value ) [inherited]
```

Setter for [editOnGesture\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.85 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.86 setGranularity()

```
setGranularity (
    value ) [inherited]
```

Setter for [granularity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.87 setGridVisible()

```
setGridVisible (
    value ) [inherited]
```

Setter for [gridVisible\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.88 setHeadersource()

```
setHeadersource (
    value ) [inherited]
```

Setter for [headersource\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.89 setHideExpanders()

```
setHideExpanders (
    value ) [inherited]
```

Setter for [hideExpanders\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.90 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.91 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.92 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.93 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.94 setNodeActivationNeedsDoubleClick()

```
setNodeActivationNeedsDoubleClick (
    value ) [inherited]
```

Setter for [nodeActivationNeedsDoubleClick\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.95 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.57.2.96 `setRowsResizable()`

```
setRowsResizable (
    value ) [inherited]
```

Setter for [rowsResizable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.97 `setShowChangeMenu()`

```
setShowChangeMenu (
    value ) [inherited]
```

Setter for [showChangeMenu\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.98 `setShowOnlyFirstColumn()`

```
setShowOnlyFirstColumn (
    value ) [inherited]
```

Setter for [showOnlyFirstColumn\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.99 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.100 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.101 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.102 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.103 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.57.2.104 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.105 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.106 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.57.2.107 showChangeMenu()

```
showChangeMenu ( ) [inherited]
```

If a menu shall be shown for making manual changes to the data source.

Instead of true, it may also be a configuration object, which may contain the following:

- `item_foo_label`, `item_foo_icon`, `item_foo_isvisible`, `item_foo_action`: Specifies a menu action `foo` by four functions. Each function gets `widget`, `datasource` as parameters. Respectively they have to return a label, an icon, if it is actually visible, or have to execute the action. It is also possible to customize the existing actions `addrow`, `addrowbefore`, `addcolumn`, `addcolumnbefore`, `removerow`, `removecolumn` and edit this way.

1.57.2.108 showOnlyFirstColumn()

```
showOnlyFirstColumn ( ) [inherited]
```

If to show only the first column and hide the other ones.

1.57.2.109 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretch←Affinity() == 0`.

1.57.2.110 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.57.2.111 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.57.2.112 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.57.2.113 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.57.2.114 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.57.2.115 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.57.2.116 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

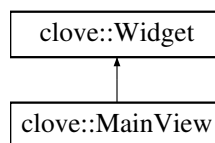
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.58 clove::MainView Class Reference

A typical main view, including a [clove::Toolbar](#) and a container for an inner body widget.

Inheritance diagram for [clove::MainView](#):



Public Member Functions

- [sidebar](#) ()
The sidebar widget as widget configuration like for [clove::build\(\)](#).
- [setSidebar](#) (value)
Setter for [sidebar\(\)](#).
- [bodyLeft](#) ()
The left inner widget as widget configuration like for [clove::build\(\)](#).
- [setBodyLeft](#) (value)
Setter for [bodyLeft\(\)](#).
- [bodyRight](#) ()
The right inner widget as widget configuration like for [clove::build\(\)](#).
- [setBodyRight](#) (value)
Setter for [bodyRight\(\)](#).
- [bodyLeftViewActionLabel](#) ()
The title string for the left inner widget.
- [setBodyLeftViewActionLabel](#) (value)

- Setter for [bodyLeftViewActionLabel\(\)](#).
- [bodyRightViewActionLabel](#) ()
 - The title string for the right inner widget.
- [setBodyRightViewActionLabel](#) (value)
 - Setter for [bodyRightViewActionLabel\(\)](#).
- [splitterPosition](#) ()
 - The position of the splitter between the left and right inner widget as number between 0.0 and 1.0.
- [setSplitterPosition](#) (value)
 - Setter for [splitterPosition\(\)](#).
- [switchToRightBody](#) ()
 - If in small mode, it switches the view to the right body.
- [switchToLeftBody](#) ()
 - If in small mode, it switches the view to the left body.
- [head1](#) ()
 - The 1st header text.
- [setHead1](#) (value)
 - Setter for [head1\(\)](#).
- [head2](#) ()
 - The 2nd header text.
- [setHead2](#) (value)
 - Setter for [head2\(\)](#).
- [headControl](#) ()
 - An optional widget for arbitrary control purposes as widget configuration like for [clove::build\(\)](#). It's displayed below the menu bar in full width.
- [setHeadControl](#) (value)
 - Setter for [headControl\(\)](#).
- [headControlWidget](#) ()
 - Returns the control widget as [clove::Widget](#) (or undefined if no [headControl\(\)](#) is set).
- [icon](#) ()
 - The header icon as [clove::Icon](#).
- [setIcon](#) (value)
 - Setter for [icon\(\)](#).
- [actions](#) ()
 - Menu actions.
- [setActions](#) (value)
 - Setter for [actions\(\)](#).
- [showOnly](#) ()
 - Specifies if to forcefully show just one of the two main panels (0, 1, undefined).
- [setShowOnly](#) (value)
 - Setter for [showOnly\(\)](#).
- [declareProperty](#) (k, defaultV)
 - Declares a widget property.
- [getProperty](#) (k)
 - General-purpose getter for widget properties.
- [setProperty](#) (k, v)
 - General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
 - Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
 - Initializes the widget.
- [doinit](#) ()

- Executes late widget initialization (i.e. after properties are applied).*

 - [doinitEarly](#) ()
- Executes early widget initialization (i.e. before properties are applied).*

 - [resize](#) ()
- Applies the new widget size to internal content.*

 - [doresize](#) ()
- Corrects alignments of internal elements according to the new widget size.*

 - [relayout](#) ()
- Notifies the parent widget that a new geometry is required.*

 - [focus](#) ()
- Sets the focus to this widget.*

 - [isAlive](#) ()
- Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).*

 - [computeMinimalWidth](#) ()
- Computes the minimal width in pixel this widget needs to have.*

 - [computePreferredWidth](#) ()
- Computes the preferred width in pixel this widget needs to have.*

 - [computeMinimalHeightForWidth](#) (w)
- Computes the minimal height in pixel this widget needs to have for a given width.*

 - [computePreferredHeightForWidth](#) (w)
- Computes the preferred height in pixel this widget needs to have for a given width.*

 - [getMinimalWidth](#) ()
- Returns the minimal width in pixel this widget needs to have.*

 - [getPreferredWidth](#) ()
- Returns the preferred width in pixel this widget needs to have.*

 - [getMinimalHeightForWidth](#) (w)
- Returns the minimal height in pixel this widget needs to have for a given width.*

 - [getPreferredHeightForWidth](#) (w)
- Returns the preferred height in pixel this widget needs to have for a given width.*

 - [addStyleClass](#) (css)
- Adds the css class `css` to this widget.*

 - [removeStyleClass](#) (css)
- Removes the css class `css` from this widget.*

 - [isStyleClass](#) (css)
- Checks if this widget contains the css class `css`.*

 - [setStyleClassAssigned](#) (css, assigned)
- Sets or unsets the css class `css` to this widget.*

 - [containingNameScope](#) ()
- The namespace this widget contains to.*

 - [nameScope](#) ()
- The own namespace (or the namespace it contains to, if this widget does not bring a new one).*

 - [remove](#) (removeconfig)
- Removes this widget.*

 - [effectivelyEnabled](#) ()
- If this widget is enabled and not marked as busy (i.e. can interact with the user).*

 - [effectiveVisibility](#) ()
- If this widget and all parent widgets are visible. See also [visibility\(\)](#).*

 - [childrenWidgets](#) ()
- List of the children `clove::Widget` instances.*

 - [parentWidget](#) ()
- The parent `clove::Widget`.*

- `name ()`
The name of the widget.
- `setName (value)`
Setter for `name()`.
- `enabled ()`
If this widget is marked as enabled (i.e. can interact with the user).
- `setEnabled (value)`
Setter for `enabled()`.
- `styleClass ()`
Custom css class(es).
- `setStyleClass (value)`
Setter for `styleClass()`.
- `style ()`
Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle (value)`
Setter for `style()`.
- `visibility ()`
If this widget is visible.
- `setVisibility (value)`
Setter for `visibility()`.
- `doStandaloneResizing ()`
If the widget handles to resize itself as needed.
- `setDoStandaloneResizing (value)`
Setter for `doStandaloneResizing()`.
- `mayFocus ()`
If the widget can have the keyboard focus.
- `setMayFocus (value)`
Setter for `mayFocus()`.
- `horizontalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity (value)`
Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity (value)`
Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing ()`
If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing (value)`
Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing ()`
If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing (value)`
Setter for `strictVerticalSizing()`.
- `vstretch ()`
Alias for `verticalStretchAffinity()`.
- `setVstretch (value)`
Setter for `vstretch()`.
- `hstretch ()`
Alias for `horizontalStretchAffinity()`.
- `setHstretch (value)`

- Setter for [hstretch\(\)](#).
 - [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.
 - [setBusy](#) (value)
 - Setter for [busy\(\)](#).
 - [registerBusy](#) ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.
 - [unregisterBusy](#) (token)
 - Drops a token and returns to normal state if no other tokens exist.
 - [hasFocus](#) ()
 - If the widget has the keyboard focus.
 - [innerSize](#) ()
 - Returns inner width and height of this widget.
 - [outerSize](#) ()
 - Returns outer width and height of this widget.
 - [OnDestroyed](#)
 - Triggered when this widget was removed somehow.
 - [OnResized](#)
 - Triggered when this widget was resized.
 - [OnVisibilityChanged](#)
 - Triggered when the visibility of this widget changed.
 - [OnPropertyChanged](#)
 - Triggered whenever a property of this widget changed its value.
 - [OnKeyDown](#)
 - Triggered when a keyboard key went down.
 - [OnKeyUp](#)
 - Triggered when a keyboard key came up.
 - [OnKeyPress](#)
 - Triggered when a keyboard key was pressed.

1.58.1 Detailed Description

A typical main view, including a [clove::Toolbar](#) and a container for an inner body widget.

1.58.2 Member Function Documentation

1.58.2.1 actions()

```
actions ( )
```

Menu actions.

See [clove::Menubar::actions\(\)](#).

1.58.2.2 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.58.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.58.2.4 bodyLeft()

```
bodyLeft ( )
```

The left inner widget as widget configuration like for [clove::build\(\)](#).

1.58.2.5 bodyLeftViewActionLabel()

```
bodyLeftViewActionLabel ( )
```

The title string for the left inner widget.

1.58.2.6 bodyRight()

```
bodyRight ( )
```

The right inner widget as widget configuration like for [clove::build\(\)](#).

1.58.2.7 bodyRightViewActionLabel()

```
bodyRightViewActionLabel ( )
```

The title string for the right inner widget.

1.58.2.8 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.58.2.9 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.58.2.10 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.58.2.11 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.58.2.12 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.58.2.13 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.58.2.14 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.58.2.15 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.58.2.16 doinit()

`doinit () [inherited]`

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.58.2.17 doinitEarly()

`doinitEarly () [inherited]`

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.58.2.18 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.58.2.19 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.58.2.20 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.58.2.21 `effectiveVisibility()`

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.58.2.22 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.58.2.23 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.58.2.24 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.58.2.25 `getMinimalWidth()`

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.58.2.26 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.58.2.27 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.58.2.28 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.58.2.29 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.58.2.30 head1()

```
head1 ( )
```

The 1st header text.

1.58.2.31 head2()

```
head2 ( )
```

The 2nd header text.

1.58.2.32 headControl()

```
headControl ( )
```

An optional widget for arbitrary control purposes as widget configuration like for [clove::build\(\)](#). It's displayed below the menu bar in full width.

1.58.2.33 headControlWidget()

```
headControlWidget ( )
```

Returns the control widget as [clove::Widget](#) (or undefined if no [headControl\(\)](#) is set).

1.58.2.34 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.58.2.35 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.58.2.36 icon()

```
icon ( )
```

The header icon as [clove::Icon](#).

1.58.2.37 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.58.2.38 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.58.2.39 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.58.2.40 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class *class*.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.58.2.41 mayFocus()

`mayFocus () [inherited]`

If the widget can have the keyboard focus.

1.58.2.42 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.58.2.43 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.58.2.44 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.58.2.45 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.58.2.46 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.58.2.47 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.58.2.48 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.58.2.49 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.58.2.50 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.58.2.51 `outerSize()`

`outerSize () [inherited]`

Returns outer width and height of this widget.

1.58.2.52 `parentWidget()`

`parentWidget () [inherited]`

The parent [Clove::Widget](#).

1.58.2.53 `registerBusy()`

`registerBusy () [inherited]`

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.58.2.54 `relayout()`

`relayout () [inherited]`

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.58.2.55 `remove()`

`remove (
 removeconfig) [inherited]`

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.58.2.56 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.58.2.57 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.58.2.58 setActions()

```
setActions (
    value )
```

Setter for [actions\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.59 setBodyLeft()

```
setBodyLeft (
    value )
```

Setter for [bodyLeft\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.60 setBodyLeftViewActionLabel()

```
setBodyLeftViewActionLabel (
    value )
```

Setter for [bodyLeftViewActionLabel\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.61 setBodyRight()

```
setBodyRight (
    value )
```

Setter for [bodyRight\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.62 setBodyRightViewActionLabel()

```
setBodyRightViewActionLabel (
    value )
```

Setter for [bodyRightViewActionLabel\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.63 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.64 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.65 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.66 setHead1()

```
setHead1 (
    value )
```

Setter for [head1\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.67 setHead2()

```
setHead2 (
    value )
```

Setter for [head2\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.68 setHeadControl()

```
setHeadControl (
    value )
```

Setter for [headControl\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.69 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.70 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.71 setIcon()

```
setIcon (
    value )
```

Setter for [icon\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.72 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.73 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.74 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.58.2.75 setShowOnly()

```
setShowOnly (
    value )
```

Setter for [showOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.76 setSidebar()

```
setSidebar (
    value )
```

Setter for [sidebar\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.77 setSplitterPosition()

```
setSplitterPosition (
    value )
```

Setter for [splitterPosition\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.78 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.79 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.80 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.81 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.82 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.58.2.83 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.84 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.85 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.58.2.86 showOnly()

```
showOnly ( )
```

Specifies if to forcefully show just one of the two main panels (0, 1, undefined).

1.58.2.87 sidebar()

```
sidebar ( )
```

The sidebar widget as widget configuration like for [clove::build\(\)](#).

1.58.2.88 splitterPosition()

```
splitterPosition ( )
```

The position of the splitter between the left and right inner widget as number between 0.0 and 1.0.

1.58.2.89 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.58.2.90 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.58.2.91 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.58.2.92 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.58.2.93 switchToLeftBody()

```
switchToLeftBody ( )
```

If in small mode, it switches the view to the left body.

1.58.2.94 switchToRightBody()

```
switchToRightBody ( )
```

If in small mode, it switches the view to the right body.

1.58.2.95 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.58.2.96 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.58.2.97 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.58.2.98 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

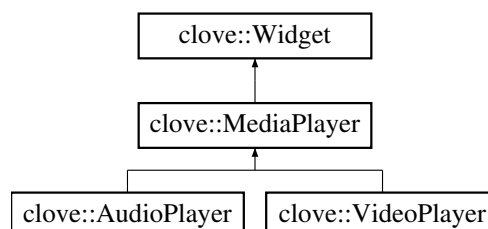
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.59 clove::MediaPlayer Class Reference

A base class for media player widgets.

Inheritance diagram for `clove::MediaPlayer`:



Public Member Functions

- `autoplay ()`
If to automatically start playback as soon as possible.
- `setAutoPlay (value)`
Setter for `autoplay()`.
- `showControls ()`
If to show user controls for play, pause and more.
- `setShowControls (value)`
Setter for `showControls()`.
- `currentMediaPosition ()`
The current media playback position in seconds.
- `setCurrentMediaPosition (value)`
Setter for `currentMediaPosition()`.
- `loop ()`
If to playback in an endless loop.
- `setLoop (value)`
Setter for `loop()`.
- `muted ()`
If to mute the audio playback.
- `setMuted (value)`
Setter for `muted()`.
- `preload ()`
If to begin loading the media data as soon as possible.
- `setPreload (value)`
Setter for `preload()`.
- `volume ()`
The audio playback volume between 0 . 0 and 1 . 0.
- `setVolume (value)`
Setter for `volume()`.
- `source ()`
The media source URL.
- `setSource (value)`
Setter for `source()`.
- `duration ()`
The duration of the loaded media source in seconds.
- `hasEnded ()`
If the playback has ended (i.e. reached the end).
- `isPaused ()`
If the playback is logically paused.
- `nativeNode ()`
Returns the native html dom node.
- `play ()`
Starts playback.
- `pause ()`
Pauses playback.
- `OnLoadingAborted`
Triggered when the media loading aborted for some reasons (e.g. network errors).
- `OnReadyForPlayback`
Triggered when there are enough data for starting playback.
- `OnDurationChanged`

- Triggered when the playback duration changed.*
- [OnHasEnded](#)
Triggered when the playback has ended.
- [OnIsPaused](#)
Triggered when the playback logically paused.
- [OnIsPlaying](#)
Triggered when the playback logically starts.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)
Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)
Removes the css class `css` from this widget.

- [isStyleClass](#) (class)

Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)

Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.
- [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)

Removes this widget.
- [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()

List of the children `clove::Widget` instances.
- [parentWidget](#) ()

The parent `clove::Widget`.
- [name](#) ()

The name of the widget.
- [setName](#) (value)

Setter for [name\(\)](#).
- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)

Setter for [enabled\(\)](#).
- [styleClass](#) ()

Custom css class(es).
- [setStyleClass](#) (value)

Setter for [styleClass\(\)](#).
- [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)

Setter for [style\(\)](#).
- [visibility](#) ()

If this widget is visible.
- [setVisibility](#) (value)

Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()

If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)

Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()

If the widget can have the keyboard focus.
- [setMayFocus](#) (value)

Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)

Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()

- Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.*
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
 - [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
 - [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
 - [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
 - [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
 - [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
 - [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
 - [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
 - [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
 - [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
 - [setBusy](#) (value)
Setter for [busy\(\)](#).
 - [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
 - [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
 - [hasFocus](#) ()
If the widget has the keyboard focus.
 - [innerSize](#) ()
Returns inner width and height of this widget.
 - [outerSize](#) ()
Returns outer width and height of this widget.
 - [OnDestroyed](#)
Triggered when this widget was removed somehow.
 - [OnResized](#)
Triggered when this widget was resized.
 - [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
 - [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
 - [OnKeyDown](#)
Triggered when a keyboard key went down.
 - [OnKeyUp](#)
Triggered when a keyboard key came up.
 - [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.59.1 Detailed Description

A base class for media player widgets.

This is an abstract class. Use one of the subclasses.

Note: This interface provides actions and events for typical usage. For other scenarios, use [nativeNode\(\)](#) for retrieving the `<audio>` or `<video>` html dom node. They provide a more powerful api.

1.59.2 Member Function Documentation

1.59.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.59.2.2 autoPlay()

```
autoPlay ( )
```

If to automatically start playback as soon as possible.

1.59.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<code>k</code>	The widget property name.
<code>vb</code>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.59.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.59.2.5 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.59.2.6 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.59.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.59.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.59.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.59.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.59.2.11 currentMediaPosition()

```
currentMediaPosition ( )
```

The current media playback position in seconds.

1.59.2.12 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.59.2.13 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.59.2.14 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.59.2.15 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.59.2.16 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.59.2.17 duration()

```
duration ( )
```

The duration of the loaded media source in seconds.

1.59.2.18 `effectivelyEnabled()`

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.59.2.19 `effectiveVisibility()`

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.59.2.20 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.59.2.21 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.59.2.22 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.59.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.59.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.59.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.59.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.59.2.27 hasEnded()

```
hasEnded ( )
```

If the playback has ended (i.e. reached the end).

1.59.2.28 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.59.2.29 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.59.2.30 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.59.2.31 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.59.2.32 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.59.2.33 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.59.2.34 isPaused()

```
isPaused ( )
```

If the playback is logically paused.

This does not return `true` just when loading stalls.

1.59.2.35 isStyleClass()

```
isStyleClass (
    css ) [inherited]
```

Checks if this widget contains the css class `css`.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.59.2.36 loop()

```
loop ( )
```

If to playback in an endless loop.

1.59.2.37 mayFocus()

`mayFocus () [inherited]`

If the widget can have the keyboard focus.

1.59.2.38 muted()

`muted ()`

If to mute the audio playback.

1.59.2.39 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.59.2.40 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.59.2.41 nativeNode()

`nativeNode ()`

Returns the native html dom node.

1.59.2.42 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.43 OnDurationChanged()

OnDurationChanged

Triggered when the playback duration changed.

See also [duration\(\)](#).

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.44 OnHasEnded()

OnHasEnded

Triggered when the playback has ended.

See also [hasEnded\(\)](#).

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.45 OnIsPaused()

OnIsPaused

Triggered when the playback logically paused.

This is not triggered when loading stalls.

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.46 OnIsPlaying()

OnIsPlaying

Triggered when the playback logically starts.

This is not triggered when loading has stalled and resumes.

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.47 OnKeyDown()

OnKeyDown [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.48 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.49 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.50 OnLoadingAborted()

`OnLoadingAborted`

Triggered when the media loading aborted for some reasons (e.g. network errors).

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.51 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.52 OnReadyForPlayback()

`OnReadyForPlayback`

Triggered when there are enough data for starting playback.

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.53 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.54 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.59.2.55 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.59.2.56 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.59.2.57 pause()

pause ()

Pauses playback.

1.59.2.58 play()

play ()

Starts playback.

1.59.2.59 preload()

preload ()

If to begin loading the media data as soon as possible.

1.59.2.60 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.59.2.61 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.59.2.62 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.59.2.63 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.59.2.64 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.59.2.65 `setAutoPlay()`

```
setAutoPlay (
    value )
```

Setter for [autoPlay\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.66 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.67 `setCurrentMediaPosition()`

```
setCurrentMediaPosition (
    value )
```

Setter for [currentMediaPosition\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.68 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.69 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.70 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.71 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.72 setLoop()

```
setLoop (
    value )
```

Setter for [loop\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.73 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.74 setMuted()

```
setMuted (
    value )
```

Setter for [muted\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.75 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.76 setPreload()

```
setPreload (
    value )
```

Setter for [preload\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.77 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.59.2.78 setShowControls()

```
setShowControls (
    value )
```

Setter for [showControls\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.79 setSource()

```
setSource (
    value )
```

Setter for [source\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.80 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.81 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.82 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for `style()`.

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.83 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for `styleClass()`.

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.84 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.59.2.85 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for `verticalStretchAffinity()`.

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.86 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.87 setVolume()

```
setVolume (
    value )
```

Setter for [volume\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.88 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.59.2.89 showControls()

```
showControls ( )
```

If to show user controls for play, pause and more.

1.59.2.90 source()

```
source ( )
```

The media source URL.

1.59.2.91 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.59.2.92 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.59.2.93 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.59.2.94 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.59.2.95 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.59.2.96 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.59.2.97 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.59.2.98 volume()

`volume ()`

The audio playback volume between 0.0 and 1.0.

1.59.2.99 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

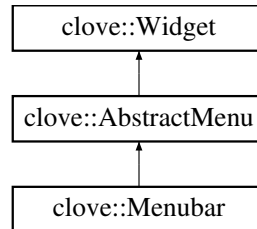
The documentation for this class was generated from the following file:

- `_meta/readme/clove.js`

1.60 clove::Menubar Class Reference

A menu bar.

Inheritance diagram for clove::Menubar:



Public Member Functions

- [actions](#) ()
The actions to be shown in this menu.
- [setActions](#) (value)
Setter for [actions\(\)](#).
- [OnActionTriggered](#)
Triggered when a menu action was chosen by the user for execution.
- [OnBeforeSubactionsExpanded](#)
Triggered just before a branch of subaction is expanded.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.

- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (class)
Adds the css class `class` to this widget.
- [removeStyleClass](#) (class)
Removes the css class `class` from this widget.
- [isStyleClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)

- Setter for `style()`.
- `visibility ()`
 - If this widget is visible.
- `setVisibility (value)`
 - Setter for `visibility()`.
- `doStandaloneResizing ()`
 - If the widget handles to resize itself as needed.
- `setDoStandaloneResizing (value)`
 - Setter for `doStandaloneResizing()`.
- `mayFocus ()`
 - If the widget can have the keyboard focus.
- `setMayFocus (value)`
 - Setter for `mayFocus()`.
- `horizontalStretchAffinity ()`
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity (value)`
 - Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity ()`
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity (value)`
 - Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing ()`
 - If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing (value)`
 - Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing ()`
 - If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing (value)`
 - Setter for `strictVerticalSizing()`.
- `vstretch ()`
 - Alias for `verticalStretchAffinity()`.
- `setVstretch (value)`
 - Setter for `vstretch()`.
- `hstretch ()`
 - Alias for `horizontalStretchAffinity()`.
- `setHstretch (value)`
 - Setter for `hstretch()`.
- `busy ()`
 - If the widget is in busy state, typically resulting in a loading animation.
- `setBusy (value)`
 - Setter for `busy()`.
- `registerBusy ()`
 - Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy (token)`
 - Drops a token and returns to normal state if no other tokens exist.
- `hasFocus ()`
 - If the widget has the keyboard focus.
- `innerSize ()`
 - Returns inner width and height of this widget.
- `outerSize ()`
 - Returns outer width and height of this widget.

- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.60.1 Detailed Description

A menu bar.

1.60.2 Member Function Documentation

1.60.2.1 `actions()`

```
actions ( ) [inherited]
```

The actions to be shown in this menu.

It is a list of action configuration objects, each having a structure like `{name:'myaction', label:'My menu action'}`. It can have a list of sub-items in the `subactions` property.

Instead of a simple list, it may also be a [clove::Datasource](#) with the action configuration structures aligned in rows.

Use [clove::MenuSeparator](#) for a separator.

One action allows this properties:

- `name`: The action name.
- `label`: The label string.
- `icon`: A [clove::Icon](#).
- `disabled`: If it is disabled.
- `invisible`: If it is invisible.
- `checkable`: If it is checkable.
- `checked`: If it is checked.

1.60.2.2 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.60.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.60.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.60.2.5 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.60.2.6 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.60.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.60.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.60.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.60.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.60.2.11 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.60.2.12 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.60.2.13 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.60.2.14 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.60.2.15 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.60.2.16 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.60.2.17 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.60.2.18 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.60.2.19 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.60.2.20 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.60.2.21 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.60.2.22 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.60.2.23 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.60.2.24 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.60.2.25 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.60.2.26 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.60.2.27 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.60.2.28 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.60.2.29 `innerSize()`

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.60.2.30 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.60.2.31 `isStyleClass()`

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.60.2.32 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.60.2.33 `name()`

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.60.2.34 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.60.2.35 OnActionTriggered()

`OnActionTriggered [inherited]`

Triggered when a menu action was chosen by the user for execution.

The event arguments contain the selected action name in `action`.

This is a [clove.Event](#) instance. See Manual for details.

1.60.2.36 OnBeforeSubactionsExpanded()

`OnBeforeSubactionsExpanded [inherited]`

Triggered just before a branch of subaction is expanded.

Implement this event for populating it dynamically.

This is a [clove.Event](#) instance. See Manual for details.

1.60.2.37 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.60.2.38 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.60.2.39 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.60.2.40 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.60.2.41 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.60.2.42 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.60.2.43 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.60.2.44 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.60.2.45 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.60.2.46 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.60.2.47 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.60.2.48 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.60.2.49 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.60.2.50 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.60.2.51 `setActions()`

```
setActions (
    value ) [inherited]
```

Setter for [actions\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.52 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.53 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.54 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.55 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.56 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.57 `setMayFocus()`

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.58 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.59 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.60.2.60 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.61 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.62 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.63 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.64 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.60.2.65 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (  
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.66 `setVisibility()`

```
setVisibility (  
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.67 `setVstretch()`

```
setVstretch (  
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.60.2.68 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity()` == 0.

1.60.2.69 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity()` == 0.

1.60.2.70 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.60.2.71 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.60.2.72 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.60.2.73 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.60.2.74 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.60.2.75 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

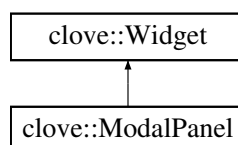
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.61 clove::ModalPanel Class Reference

A surface for implementing modality.

Inheritance diagram for `clove::ModalPanel`:



Public Member Functions

- [fullyTransparent](#) ()
If the modal panel is fully transparent (i.e. not really visible) instead of semi-transparent.
- [setFullyTransparent](#) (value)
Setter for [fullyTransparent\(\)](#).
- [OnClicked](#)
Triggered when the user clicks on this modal panel.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)
Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)

- Removes the css class `class` from this widget.*

 - `isStyleClass` (`class`)

Checks if this widget contains the css class `class`.
- `setStyleClassAssigned` (`class`, `assigned`)

Sets or unsets the css class `class` to this widget.
- `containingNameScope` ()

The namespace this widget contains to.
- `nameScope` ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- `remove` (`removeconfig`)

Removes this widget.
- `effectivelyEnabled` ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
- `effectiveVisibility` ()

If this widget and all parent widgets are visible. See also `visibility()`.
- `childrenWidgets` ()

List of the children `clove::Widget` instances.
- `parentWidget` ()

The parent `clove::Widget`.
- `name` ()

The name of the widget.
- `setName` (`value`)

Setter for `name()`.
- `enabled` ()

If this widget is marked as enabled (i.e. can interact with the user).
- `setEnabled` (`value`)

Setter for `enabled()`.
- `styleClass` ()

Custom css class(es).
- `setStyleClass` (`value`)

Setter for `styleClass()`.
- `style` ()

Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle` (`value`)

Setter for `style()`.
- `visibility` ()

If this widget is visible.
- `setVisibility` (`value`)

Setter for `visibility()`.
- `doStandaloneResizing` ()

If the widget handles to resize itself as needed.
- `setDoStandaloneResizing` (`value`)

Setter for `doStandaloneResizing()`.
- `mayFocus` ()

If the widget can have the keyboard focus.
- `setMayFocus` (`value`)

Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity` (`value`)

Setter for `horizontalStretchAffinity()`.

- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity](#)().
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing](#)().
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing](#)().
- [vstretch](#) ()
Alias for [verticalStretchAffinity](#)().
- [setVstretch](#) (value)
Setter for [vstretch](#)().
- [hstretch](#) ()
Alias for [horizontalStretchAffinity](#)().
- [setHstretch](#) (value)
Setter for [hstretch](#)().
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy](#)().
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.61.1 Detailed Description

A surface for implementing modality.

For a certain time, it will be inserted before the main user interface (i.e. higher on z-axis) in order to block user interaction with it.

It is typically not required to use it directly. Try `clove::Dialog::show()` and `clove::utils::popup()` before.

1.61.2 Member Function Documentation

1.61.2.1 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.61.2.2 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<code>k</code>	The widget property name.
<code>vb</code>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.61.2.3 `busy()`

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.61.2.4 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.61.2.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.61.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.61.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.61.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.61.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.61.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.61.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.61.2.12 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.61.2.13 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.61.2.14 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.61.2.15 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.61.2.16 effectiveVisibility()

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.61.2.17 enabled()

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.61.2.18 focus()

`focus () [inherited]`

Sets the focus to this widget.

1.61.2.19 fullyTransparent()

```
fullyTransparent ( )
```

If the modal panel is fully transparent (i.e. not really visible) instead of semi-transparent.

1.61.2.20 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.61.2.21 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.61.2.22 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.61.2.23 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.61.2.24 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.61.2.25 hasFocus()

`hasFocus () [inherited]`

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.61.2.26 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.61.2.27 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.61.2.28 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.61.2.29 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.61.2.30 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.61.2.31 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.61.2.32 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.61.2.33 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.61.2.34 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.61.2.35 OnClicked()

`OnClicked`

Triggered when the user clicks on this modal panel.

This is a [clove.Event](#) instance. See Manual for details.

1.61.2.36 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.61.2.37 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.61.2.38 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.61.2.39 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.61.2.40 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.61.2.41 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.61.2.42 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.61.2.43 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.61.2.44 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.61.2.45 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.61.2.46 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.61.2.47 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.61.2.48 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.61.2.49 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.61.2.50 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.51 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.52 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.53 setFullyTransparent()

```
setFullyTransparent (
    value )
```

Setter for [fullyTransparent\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.54 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (  
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.55 setHstretch()

```
setHstretch (  
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.56 setMayFocus()

```
setMayFocus (  
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.57 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.58 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.61.2.59 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.60 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.61 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.62 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.63 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.61.2.64 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.65 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.66 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.61.2.67 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.61.2.68 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.61.2.69 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.61.2.70 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.61.2.71 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.61.2.72 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.61.2.73 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.61.2.74 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

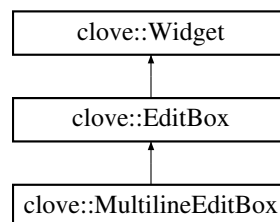
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.62 clove::MultilineEditBox Class Reference

A multi-line box for text input from the user.

Inheritance diagram for clove::MultilineEditBox:



Public Member Functions

- [readOnly](#) ()
If the edit box is read-only or editable by the user.
- [setReadOnly](#) (value)
Setter for [readOnly\(\)](#).
- [text](#) ()
The current text.
- [setText](#) (value)
Setter for [text\(\)](#).
- [hintText](#) ()
The hint text.
- [setHintText](#) (value)
Setter for [hintText\(\)](#).
- [autocompleteItems](#) ()

- A [Clove.DataSource](#) with a list of autocompletion items to popup.
- [setAutocompletionItems](#) (value)

Setter for [autocompletionItems\(\)](#).
- [autocompletionFilter](#) ()

How to filter the autocompletion list.
- [setAutocompletionFilter](#) (value)

Setter for [autocompletionFilter\(\)](#).
- [autocompletionOpenForNoText](#) ()

If to show the autocompletion list when the edit box is empty.
- [setAutocompletionOpenForNoText](#) (value)

Setter for [autocompletionOpenForNoText\(\)](#).
- [setTextSelection](#) (ifrom, ito)

Selects the specified part of the text.
- [textSelection](#) ()

Returns the current text selection indices.
- [OnChanged](#)

Triggered when the text was changed.
- [OnPopupTextSelected](#)

Triggered when a text was selected from the popup.
- [declareProperty](#) (k, defaultV)

Declares a widget property.
- [getProperty](#) (k)

General-purpose getter for widget properties.
- [setProperty](#) (k, v)

General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)

Initializes the widget.
- [doinit](#) ()

Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()

Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()

Applies the new widget size to internal content.
- [doresize](#) ()

Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()

Notifies the parent widget that a new geometry is required.
- [focus](#) ()

Sets the focus to this widget.
- [isAlive](#) ()

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()

Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()

Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)

Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)

Computes the preferred height in pixel this widget needs to have for a given width.

- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (clss)
Adds the css class `clss` to this widget.
- [removeStyleClass](#) (clss)
Removes the css class `clss` from this widget.
- [isStyleClass](#) (clss)
Checks if this widget contains the css class `clss`.
- [setStyleClassAssigned](#) (clss, assigned)
Sets or unsets the css class `clss` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()

- If the widget handles to resize itself as needed.*

 - [setDoStandaloneResizing](#) (value)
 Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
 Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
 Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
 Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
 Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
 Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
 Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.

- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.62.1 Detailed Description

A multi-line box for text input from the user.

1.62.2 Member Function Documentation

1.62.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.62.2.2 autoCompleteFilter()

```
autoCompleteFilter ( ) [inherited]
```

How to filter the autocomplete list.

Either `'startswith', 'contains', function(itemtext, boxtext)` or undefined.

1.62.2.3 autoCompleteItems()

```
autoCompleteItems ( ) [inherited]
```

A [clove.Datasource](#) with a list of autocomplete items to popup.

It is allowed to observe the `text` in order to generate live content for this list.

1.62.2.4 autoCompleteOpenForNoText()

```
autoCompletionOpenForNoText ( ) [inherited]
```

If to show the autocomplete list when the edit box is empty.

1.62.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.62.2.6 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.62.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.62.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.62.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.62.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.62.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.62.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.62.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.62.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.62.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.62.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.62.2.17 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.62.2.18 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.62.2.19 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.62.2.20 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.62.2.21 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.62.2.22 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.62.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.62.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.62.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.62.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.62.2.27 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.62.2.28 hintText()

```
hintText ( ) [inherited]
```

The hint text.

Typically contains something like a label text. It is shown as a hint when the box is empty.

1.62.2.29 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.62.2.30 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.62.2.31 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.62.2.32 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.62.2.33 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.62.2.34 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.62.2.35 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.62.2.36 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.62.2.37 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.62.2.38 OnChanged()

`OnChanged [inherited]`

Triggered when the text was changed.

This is a [clove.Event](#) instance. See Manual for details.

1.62.2.39 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.62.2.40 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.62.2.41 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.62.2.42 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.62.2.43 OnPopupTextSelected()

OnPopupTextSelected [inherited]

Triggered when a text was selected from the popup.

This is a [clove.Event](#) instance. See Manual for details.

1.62.2.44 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.62.2.45 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.62.2.46 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.62.2.47 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.62.2.48 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.62.2.49 readOnly()

```
readOnly ( ) [inherited]
```

If the edit box is read-only or editable by the user.

1.62.2.50 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.62.2.51 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.62.2.52 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.62.2.53 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.62.2.54 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.62.2.55 `setAutocompletionFilter()`

```
setAutocompletionFilter (
    value ) [inherited]
```

Setter for [autocompletionFilter\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.56 `setAutocompletionItems()`

```
setAutocompletionItems (
    value ) [inherited]
```

Setter for [autocompletionItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.57 `setAutocompletionOpenForNoText()`

```
setAutocompletionOpenForNoText (
    value ) [inherited]
```

Setter for [autocompletionOpenForNoText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.58 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.59 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.60 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.61 `setHintText()`

```
setHintText (
    value ) [inherited]
```

Setter for [hintText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.62 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.63 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.64 `setMayFocus()`

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.65 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.66 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.62.2.67 setReadOnly()

```
setReadOnly (
    value ) [inherited]
```

Setter for [readOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.68 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (  
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.69 setStrictVerticalSizing()

```
setStrictVerticalSizing (  
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.70 setStyle()

```
setStyle (  
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.71 setStyleClass()

```
setStyleClass (  
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.72 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.62.2.73 `setText()`

```
setText (
    value ) [inherited]
```

Setter for [text\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.74 `setTextSelection()`

```
setTextSelection (
    ifrom,
    ito ) [inherited]
```

Selects the specified part of the text.

Parameters

<i>ifrom</i>	The selection begin index.
<i>ito</i>	The selection end index.

1.62.2.75 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.76 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.77 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.62.2.78 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.62.2.79 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.62.2.80 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.62.2.81 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.62.2.82 text()

```
text ( ) [inherited]
```

The current text.

1.62.2.83 textSelection()

```
textSelection ( ) [inherited]
```

Returns the current text selection indices.

1.62.2.84 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.62.2.85 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.62.2.86 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.62.2.87 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

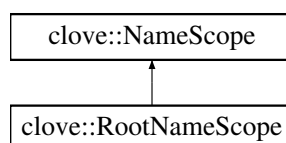
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.63 clove::NameScope Class Reference

A mixin realizing a new namespace.

Inheritance diagram for [clove::NameScope](#):



Public Member Functions

- [getName](#) (name)
Finds a widget by name within this namespace.
- [addChildNameScope](#) (childnamespace)
Adds a namespace to the children of this one, so a searcher will also search in this child.

1.63.1 Detailed Description

A mixin realizing a new namespace.

Read the Manual for details.

1.63.2 Member Function Documentation

1.63.2.1 addChildNameScope()

```
addChildNameScope (  
    childnamespace )
```

Adds a namespace to the children of this one, so a searcher will also search in this child.

Parameters

<i>childnamespace</i>	The child namespace.
-----------------------	----------------------

1.63.2.2 getName()

```
getName (  
    name )
```

Finds a widget by name within this namespace.

Parameters

<i>name</i>	The widget name.
-------------	------------------

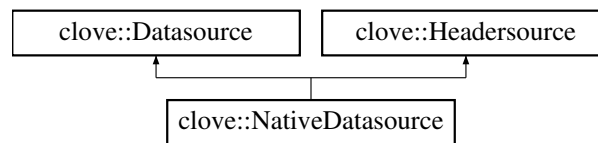
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.64 clove::NativeDatasource Class Reference

A [clove::Datasource](#) implementation which stores all data in-memory.

Inheritance diagram for `clove::NativeDatasource`:



Public Member Functions

- [NativeDatasource](#) (config)
- [root](#) ()
 - The root node as `clove::NativeDatasourceValue`.*
- [valuePointerToNativeNode](#) (ptr)
 - Returns the `clove::NativeDatasourceValue` for a node.*
- [insertRow](#) (ir, parent)
 - Inserts a new empty row.*
- [insertColumn](#) (ic, parent)
 - Inserts a new empty column.*
- [appendRow](#) (parent)
 - Appends a new empty row (like inserting to the end).*
- [appendColumn](#) (parent)
 - Appends a new empty column (like inserting to the end).*
- [removeRow](#) (ir, parent)
 - Removes a row.*
- [removeColumn](#) (ic, parent)
 - Removes a column.*
- [setValue](#) (ptr, value)
 - Sets a new value to a node.*
- [setRootValue](#) (value)
 - Sets a new value to the root node.*
- [setMetadata](#) (ptr, key, value)
 - Sets a metadata key for a node.*
- [setRowHeader](#) (irow, parent, headercfg)
 - Sets a row header configuration.*
- [setColumnHeader](#) (icol, parent, headercfg)
 - Sets a column header configuration.*
- [getValue](#) (ptr)
 - Returns the value for a given node.*
- [getMetadata](#) (ptr)
 - Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.*
- [changeValue](#) (ptr, value)
 - Change the value for a given node.*
- [rowCount](#) (parent)
 - Returns the number of rows for a given node.*
- [columnCount](#) (parent)

- Returns the number of columns for a given node.*

 - [valuePointer](#) (irow, icol, [parent](#))

Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.
- [parent](#) (ptr)

Returns the parent node for a given node.
- [valuePointerNavigateInDepth](#) (ptr, direction, mayexpandfct)

Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.
- [OnDataInsert](#)

Triggered when a node insertion takes place.
- [OnDataRemove](#)

Triggered when a node removal takes place.
- [OnDataUpdate](#)

Triggered when a node data update takes place.
- [getRowHeader](#) (irow, [parent](#))

Returns the header configuration for a given row.
- [getColumnHeader](#) (irow, [parent](#))

Returns the header configuration for a given column.
- [rowHeadersVisible](#) ([parent](#))

If row headers are visible.
- [columnHeadersVisible](#) ([parent](#))

If column headers are visible.
- [OnHeaderDataInsert](#)

Triggered when a new row or column of header data was inserted.
- [OnHeaderDataRemove](#)

Triggered when a row or column of header data was removed.
- [OnHeaderDataUpdate](#)

Triggered when header data were updated.
- [OnHeaderVisibilityUpdated](#)

Triggered when header visibilities changed.

1.64.1 Detailed Description

A [clove::Datasource](#) implementation which stores all data in-memory.

It provides an interface for populating it with data and can directly be constructed and used. It also implements [clove::Headersource](#), so it can also carry row and column header configurations.

1.64.2 Constructor & Destructor Documentation

1.64.2.1 NativeDatasource()

```
NativeDatasource (
    config )
```

The configuration object allows the following keys:

- `readonly`: If this datasource may be changed from outside.
- `rowHeadersVisible`: If row headers shall be visible in presentation.
- `columnHeadersVisible`: If column headers shall be visible in presentation.

Parameters

<i>config</i>	A configuration object.
---------------	-------------------------

1.64.3 Member Function Documentation**1.64.3.1 appendColumn()**

```
appendColumn (  
    parent )
```

Appends a new empty column (like inserting to the end).

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.64.3.2 appendRow()

```
appendRow (  
    parent )
```

Appends a new empty row (like inserting to the end).

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.64.3.3 changeValue()

```
changeValue (  
    ptr,  
    value ) [pure virtual], [inherited]
```

Change the value for a given node.

This method is called from external places, e.g. when a user makes changes in a datasource-connected widget.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>value</i>	The new value.

1.64.3.4 columnCount()

```
columnCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of columns for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.64.3.5 columnHeadersVisible()

```
columnHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If column headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.64.3.6 getColumnHeader()

```
getColumnHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given column.

Parameters

<i>irow</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.64.3.7 getMetadata()

```
getMetadata (
    ptr ) [pure virtual], [inherited]
```

Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.64.3.8 getRowHeader()

```
getRowHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given row.

Parameters

<i>irow</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.64.3.9 getValue()

```
getValue (
    ptr ) [pure virtual], [inherited]
```

Returns the value for a given node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.64.3.10 insertColumn()

```
insertColumn (
    ic,
    parent )
```

Inserts a new empty column.

Parameters

<i>ic</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.64.3.11 insertRow()

```
insertRow (
    ir,
    parent )
```

Inserts a new empty row.

Parameters

<i>ir</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.64.3.12 OnDataInsert()

```
OnDataInsert [inherited]
```

Triggered when a node insertion takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.64.3.13 OnDataRemove()

```
OnDataRemove [inherited]
```

Triggered when a node removal takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.64.3.14 OnDataUpdate()

```
OnDataUpdate [inherited]
```

Triggered when a node data update takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.64.3.15 OnHeaderDataInsert()

`OnHeaderDataInsert` [inherited]

Triggered when a new row or column of header data was inserted.

This is a [clove.Event](#) instance. See Manual for details.

1.64.3.16 OnHeaderDataRemove()

`OnHeaderDataRemove` [inherited]

Triggered when a row or column of header data was removed.

This is a [clove.Event](#) instance. See Manual for details.

1.64.3.17 OnHeaderDataUpdate()

`OnHeaderDataUpdate` [inherited]

Triggered when header data were updated.

This is a [clove.Event](#) instance. See Manual for details.

1.64.3.18 OnHeaderVisibilityUpdated()

`OnHeaderVisibilityUpdated` [inherited]

Triggered when header visibilities changed.

This is a [clove.Event](#) instance. See Manual for details.

1.64.3.19 parent()

```
parent (
    ptr ) [pure virtual], [inherited]
```

Returns the parent node for a given node.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
------------	---

1.64.3.20 removeColumn()

```
removeColumn (
```

```
    ic,  
    parent )
```

Removes a column.

Parameters

<i>ic</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.64.3.21 removeRow()

```
removeRow (   
    ir,  
    parent )
```

Removes a row.

Parameters

<i>ir</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.64.3.22 root()

```
root ( )
```

The root node as [clove::NativeDatasourceValue](#).

1.64.3.23 rowCount()

```
rowCount (   
    parent ) [pure virtual], [inherited]
```

Returns the number of rows for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.64.3.24 rowHeadersVisible()

```
rowHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If row headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.64.3.25 setColumnHeader()

```
setColumnHeader (
    icol,
    parent,
    headercfg )
```

Sets a column header configuration.

Parameters

<i>icol</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .
<i>headercfg</i>	The header configuration.

1.64.3.26 setMetadata()

```
setMetadata (
    ptr,
    key,
    value )
```

Sets a metadata key for a node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>key</i>	The metadata key, like 'icon'.
<i>value</i>	The new value.

1.64.3.27 setRootValue()

```
setRootValue (
    value )
```

Sets a new value to the root node.

This is just a shorter variant of `setValue(undefined, value)` (for scalar usage).

Parameters

<i>value</i>	The new node value.
--------------	---------------------

1.64.3.28 setRowHeader()

```
setRowHeader (
    irow,
    parent,
    headercfg )
```

Sets a row header configuration.

Parameters

<i>irow</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .
<i>headercfg</i>	The header configuration.

1.64.3.29 setValue()

```
setValue (
    ptr,
    value )
```

Sets a new value to a node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>value</i>	The new node value.

1.64.3.30 `valuePointer()`

```
valuePointer (
    irow,
    icol,
    parent ) [pure virtual], [inherited]
```

Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.

Parameters

<i>irow</i>	The row index.
<i>icol</i>	The column index.
<i>parent</i>	The parent as clove::DatasourceValuePointer .

1.64.3.31 `valuePointerNavigateInDepth()`

```
valuePointerNavigateInDepth (
    ptr,
    direction,
    mayexpandfct ) [inherited]
```

Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
<i>direction</i>	The direction (+1 or -1).
<i>mayexpandfct</i>	A function (<i>ptr</i>) which returns <code>true</code> iff this node's children are to be traversed.

1.64.3.32 `valuePointerToNativeNode()`

```
valuePointerToNativeNode (
    ptr )
```

Returns the [clove::NativeDatasourceValue](#) for a node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.65 clove::NativeDataSourceNode Class Reference

A node value implementation for [clove::NativeDataSource](#).

Public Member Functions

- [rowCount](#) ()
Returns the number of children rows.
- [columnCount](#) ()
Returns the number of children columns.
- [toRow](#) (ir)
Returns the [clove::NativeDataSourceNode](#) which is in the same column as this one, but in a different row.
- [toColumn](#) (ic)
Returns the [clove::NativeDataSourceNode](#) which is in the same row as this one, but in a different column.
- [valuePointer](#) ()
Returns a [clove::DataSourceValuePointer](#) pointing to this node.
- [getValue](#) ()
Returns the node value.
- [getMetadata](#) ()
Returns the node metadata.
- [setValue](#) (v)
Sets the node value.
- [setMetadata](#) (k, v)
Sets a node metadata value.
- [removeRow](#) (ir)
Removes a row in this node.
- [removeColumn](#) (ic)
Removes a column in this node.
- [addRow](#) (vals)
Adds a row to this node.
- [addColumn](#) (vals)
Adds a column to this node.
- [insertRow](#) (i, vals)
Inserts a row to this node.
- [insertColumn](#) (i, vals)
Inserts a column to this node.
- [getChild](#) (irow, icol)
Returns a child [clove::NativeDataSourceNode](#).

1.65.1 Detailed Description

A node value implementation for [clove::NativeDataSource](#).

Not intended for direct construction!

1.65.2 Member Function Documentation

1.65.2.1 addColumn()

```
addColumn (
    vals )
```

Adds a column to this node.

Parameters

<i>vals</i>	A list of new values (one for each row).
-------------	--

1.65.2.2 addRow()

```
addRow (
    vals )
```

Adds a row to this node.

Parameters

<i>vals</i>	A list of new values (one for each column).
-------------	---

1.65.2.3 columnCount()

```
columnCount ( )
```

Returns the number of children columns.

1.65.2.4 getChild()

```
getChild (
    irow,
    icol )
```

Returns a child [clove::NativeDatasourceNode](#).

Parameters

<i>irow</i>	The row index.
<i>icol</i>	The column index.

1.65.2.5 getMetadata()

```
getMetadata ( )
```

Returns the node metadata.

See also [clove::Datasource::getMetadata\(\)](#).

1.65.2.6 `getValue()`

```
getValue ( )
```

Returns the node value.

1.65.2.7 `insertColumn()`

```
insertColumn (
    i,
    vals )
```

Inserts a column to this node.

Parameters

<i>i</i>	The insertion position.
<i>vals</i>	A list of new values (one for each row).

1.65.2.8 `insertRow()`

```
insertRow (
    i,
    vals )
```

Inserts a row to this node.

Parameters

<i>i</i>	The insertion position.
<i>vals</i>	A list of new values (one for each column).

1.65.2.9 `removeColumn()`

```
removeColumn (
    ic )
```

Removes a column in this node.

Parameters

<i>ic</i>	The column index.
-----------	-------------------

1.65.2.10 removeRow()

```
removeRow (
    ir )
```

Removes a row in this node.

Parameters

<i>ir</i>	The row index.
-----------	----------------

1.65.2.11 rowCount()

```
rowCount ( )
```

Returns the number of children rows.

1.65.2.12 setMetadata()

```
setMetadata (
    k,
    v )
```

Sets a node metadata value.

Parameters

<i>k</i>	The metadata key, like 'icon'.
<i>v</i>	The new value.

1.65.2.13 setValue()

```
setValue (
    v )
```

Sets the node value.

Parameters

<i>v</i>	The new value.
----------	----------------

1.65.2.14 toColumn()

```
toColumn (
    ic )
```

Returns the [clove::NativeDatasourceNode](#) which is in the same row as this one, but in a different column.

Parameters

<i>ic</i>	The column index.
-----------	-------------------

1.65.2.15 toRow()

```
toRow (
    ir )
```

Returns the [clove::NativeDatasourceNode](#) which is in the same column as this one, but in a different row.

Parameters

<i>ir</i>	The row index.
-----------	----------------

1.65.2.16 valuePointer()

```
valuePointer ( )
```

Returns a [clove::DatasourceValuePointer](#) pointing to this node.

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.66 [clove::NotificationController](#) Class Reference

Controller for clove notifications.

Public Member Functions

- [notify](#) (config, notificationConfig)
Opens a new notification.

1.66.1 Detailed Description

Controller for clove notifications.

Notifications are small popups, typically in the top/right corner of the window. They can inform the user about some status changes or provide some ways for user interactions.

Use [clove::notifications](#).

1.66.2 Member Function Documentation

1.66.2.1 `notify()`

```
notify (
    config,
    notificationConfig )
```

Opens a new notification.

`notificationConfig` may contain:

- `timeout`: Closes the notification automatically after this amount of seconds.
- `closeable`: If to provide a close button.
- `priority`: Controls stuff like ordering. Default is 0, higher values mean more importance.

Parameters

<i>config</i>	A widget configuration for the notification body, as for clove::build() .
<i>notificationConfig</i>	Some configuration aspects about the notification.

Returns

The [clove::Widget](#) implementing the notification. Use it for operations like removal or getting subwidget.
This widget spans a new [clove::NameScope](#)!

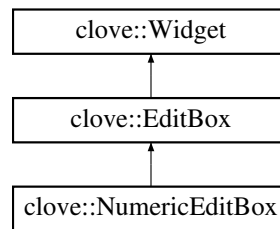
The documentation for this class was generated from the following file:

- `_meta/readme/clove.js`

1.67 `clove::NumericEditBox` Class Reference

A text box with up/down buttons for numbers.

Inheritance diagram for `clove::NumericEditBox`:



Public Member Functions

- [min](#) ()
The minimum value.
- [setMin](#) (value)
Setter for [min\(\)](#).
- [max](#) ()
The maximum value.
- [setMax](#) (value)
Setter for [max\(\)](#).
- [stepsize](#) ()
The step size.
- [setStepsize](#) (value)
Setter for [stepsize\(\)](#).
- [value](#) ()
The current numeric value.
- [setValue](#) (value)
Setter for [value\(\)](#).
- [readOnly](#) ()
If the edit box is read-only or editable by the user.
- [setReadOnly](#) (value)
Setter for [readOnly\(\)](#).
- [text](#) ()
The current text.
- [setText](#) (value)
Setter for [text\(\)](#).
- [hintText](#) ()
The hint text.
- [setHintText](#) (value)
Setter for [hintText\(\)](#).
- [autocompleteItems](#) ()
A [Clove.Datasource](#) with a list of autocomplete items to popup.
- [setAutocompleteItems](#) (value)
Setter for [autocompleteItems\(\)](#).
- [autocompleteFilter](#) ()
How to filter the autocomplete list.
- [setAutocompleteFilter](#) (value)
Setter for [autocompleteFilter\(\)](#).
- [autocompleteOpenForNoText](#) ()
If to show the autocomplete list when the edit box is empty.
- [setAutocompleteOpenForNoText](#) (value)
Setter for [autocompleteOpenForNoText\(\)](#).

- [setTextSelection](#) (ifrom, ito)
Selects the specified part of the text.
- [textSelection](#) ()
Returns the current text selection indices.
- [OnChanged](#)
Triggered when the text was changed.
- [OnPopupTextSelected](#)
Triggered when a text was selected from the popup.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (clss)
Adds the css class `clss` to this widget.
- [removeStyleClass](#) (clss)

- Removes the css class `class` from this widget.*

 - `isStyleClass` (`class`)

Checks if this widget contains the css class `class`.
- `setStyleClassAssigned` (`class`, `assigned`)

Sets or unsets the css class `class` to this widget.
- `containingNameScope` ()

The namespace this widget contains to.
- `nameScope` ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- `remove` (`removeconfig`)

Removes this widget.
- `effectivelyEnabled` ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
- `effectiveVisibility` ()

If this widget and all parent widgets are visible. See also `visibility()`.
- `childrenWidgets` ()

List of the children `clove::Widget` instances.
- `parentWidget` ()

The parent `clove::Widget`.
- `name` ()

The name of the widget.
- `setName` (`value`)

Setter for `name()`.
- `enabled` ()

If this widget is marked as enabled (i.e. can interact with the user).
- `setEnabled` (`value`)

Setter for `enabled()`.
- `styleClass` ()

Custom css class(es).
- `setStyleClass` (`value`)

Setter for `styleClass()`.
- `style` ()

Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle` (`value`)

Setter for `style()`.
- `visibility` ()

If this widget is visible.
- `setVisibility` (`value`)

Setter for `visibility()`.
- `doStandaloneResizing` ()

If the widget handles to resize itself as needed.
- `setDoStandaloneResizing` (`value`)

Setter for `doStandaloneResizing()`.
- `mayFocus` ()

If the widget can have the keyboard focus.
- `setMayFocus` (`value`)

Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity` (`value`)

Setter for `horizontalStretchAffinity()`.

- [verticalStretchAffinity \(\)](#)
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity \(value\)](#)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing \(\)](#)
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing \(value\)](#)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing \(\)](#)
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing \(value\)](#)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch \(\)](#)
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch \(value\)](#)
Setter for [vstretch\(\)](#).
- [hstretch \(\)](#)
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch \(value\)](#)
Setter for [hstretch\(\)](#).
- [busy \(\)](#)
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy \(value\)](#)
Setter for [busy\(\)](#).
- [registerBusy \(\)](#)
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy \(token\)](#)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus \(\)](#)
If the widget has the keyboard focus.
- [innerSize \(\)](#)
Returns inner width and height of this widget.
- [outerSize \(\)](#)
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.67.1 Detailed Description

A text box with up/down buttons for numbers.

1.67.2 Member Function Documentation

1.67.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.67.2.2 autoCompleteFilter()

```
autoCompleteFilter ( ) [inherited]
```

How to filter the autocomplete list.

Either 'startswith', 'contains', `function(itemtext, boxttext)` or undefined.

1.67.2.3 autoCompleteItems()

```
autoCompleteItems ( ) [inherited]
```

A [Clove.Datasource](#) with a list of autocomplete items to popup.

It is allowed to observe the `text` in order to generate live content for this list.

1.67.2.4 autoCompleteOpenForNoText()

```
autoCompleteOpenForNoText ( ) [inherited]
```

If to show the autocomplete list when the edit box is empty.

1.67.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.67.2.6 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.67.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.67.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.67.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.67.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.67.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.67.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.67.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.67.2.14 doinit()

`doinit () [inherited]`

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.67.2.15 doinitEarly()

`doinitEarly () [inherited]`

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.67.2.16 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.67.2.17 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.67.2.18 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.67.2.19 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.67.2.20 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.67.2.21 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.67.2.22 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.67.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.67.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.67.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.67.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.67.2.27 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.67.2.28 hintText()

```
hintText ( ) [inherited]
```

The hint text.

Typically contains something like a label text. It is shown as a hint when the box is empty.

1.67.2.29 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.67.2.30 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.67.2.31 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.67.2.32 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.67.2.33 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.67.2.34 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.67.2.35 max()

```
max ( )
```

The maximum value.

1.67.2.36 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.67.2.37 min()

```
min ( )
```

The minimum value.

1.67.2.38 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.67.2.39 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.67.2.40 OnChanged()

`OnChanged [inherited]`

Triggered when the text was changed.

This is a [clove.Event](#) instance. See Manual for details.

1.67.2.41 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.67.2.42 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.67.2.43 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.67.2.44 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.67.2.45 OnPopupTextSelected()

OnPopupTextSelected [inherited]

Triggered when a text was selected from the popup.

This is a [clove.Event](#) instance. See Manual for details.

1.67.2.46 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.67.2.47 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.67.2.48 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.67.2.49 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.67.2.50 `parentWidget()`

```
parentWidget ( ) [inherited]
```

The parent [clove::Widget](#).

1.67.2.51 `readOnly()`

```
readOnly ( ) [inherited]
```

If the edit box is read-only or editable by the user.

1.67.2.52 `registerBusy()`

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.67.2.53 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.67.2.54 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.67.2.55 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.67.2.56 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.67.2.57 setAutocompletionFilter()

```
setAutocompletionFilter (
    value ) [inherited]
```

Setter for [autocompletionFilter\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.58 setAutocompletionItems()

```
setAutocompletionItems (
    value ) [inherited]
```

Setter for [autocompletionItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.59 setAutocompletionOpenForNoText()

```
setAutocompletionOpenForNoText (
    value ) [inherited]
```

Setter for [autocompletionOpenForNoText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.60 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.61 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.62 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.63 `setHintText()`

```
setHintText (
    value ) [inherited]
```

Setter for [hintText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.64 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.65 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.66 setMax()

```
setMax (
    value )
```

Setter for [max\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.67 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.68 setMin()

```
setMin (
    value )
```

Setter for [min\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.69 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.70 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.67.2.71 setReadOnly()

```
setReadOnly (
    value ) [inherited]
```

Setter for [readOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.72 setStepsize()

```
setStepsize (
    value )
```

Setter for [stepsize\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.73 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.74 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.75 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.76 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.77 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.67.2.78 `setText()`

```
setText (
    value ) [inherited]
```

Setter for [text\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.79 `setTextSelection()`

```
setTextSelection (
    ifrom,
    ito ) [inherited]
```

Selects the specified part of the text.

Parameters

<i>ifrom</i>	The selection begin index.
<i>ito</i>	The selection end index.

1.67.2.80 setValue()

```
setValue (
    value )
```

Setter for [value\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.81 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.82 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.83 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.67.2.84 stepsize()

```
stepsize ( )
```

The step size.

1.67.2.85 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.67.2.86 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.67.2.87 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.67.2.88 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.67.2.89 text()

```
text ( ) [inherited]
```

The current text.

1.67.2.90 textSelection()

```
textSelection ( ) [inherited]
```

Returns the current text selection indices.

1.67.2.91 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.67.2.92 value()

```
value ( )
```

The current numeric value.

1.67.2.93 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.67.2.94 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.67.2.95 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

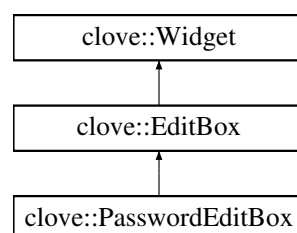
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.68 clove::PasswordEditBox Class Reference

A box for password input from the user.

Inheritance diagram for `clove::PasswordEditBox`:



Public Member Functions

- [readOnly](#) ()
If the edit box is read-only or editable by the user.
- [setReadOnly](#) (value)
Setter for [readOnly\(\)](#).
- [text](#) ()
The current text.
- [setText](#) (value)
Setter for [text\(\)](#).
- [hintText](#) ()
The hint text.
- [setHintText](#) (value)
Setter for [hintText\(\)](#).
- [autocompleteItems](#) ()
A [Clove.DataSource](#) with a list of autocomplete items to popup.
- [setAutocompleteItems](#) (value)
Setter for [autocompleteItems\(\)](#).
- [autocompleteFilter](#) ()
How to filter the autocomplete list.
- [setAutocompleteFilter](#) (value)
Setter for [autocompleteFilter\(\)](#).
- [autocompleteOpenForNoText](#) ()
If to show the autocomplete list when the edit box is empty.
- [setAutocompleteOpenForNoText](#) (value)
Setter for [autocompleteOpenForNoText\(\)](#).
- [setTextSelection](#) (ifrom, ito)
Selects the specified part of the text.
- [textSelection](#) ()
Returns the current text selection indices.
- [OnChanged](#)
Triggered when the text was changed.
- [OnPopupTextSelected](#)
Triggered when a text was selected from the popup.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()

- Corrects alignments of internal elements according to the new widget size.*

 - [relayout](#) ()

Notifies the parent widget that a new geometry is required.
- [focus](#) ()

Sets the focus to this widget.
- [isAlive](#) ()

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()

Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()

Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)

Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)

Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()

Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()

Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)

Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)

Removes the css class `css` from this widget.
- [isStyleClass](#) (css)

Checks if this widget contains the css class `css`.
- [setStyleClassAssigned](#) (css, assigned)

Sets or unsets the css class `css` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.
- [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)

Removes this widget.
- [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()

List of the children `clove::Widget` instances.
- [parentWidget](#) ()

The parent `clove::Widget`.
- [name](#) ()

The name of the widget.
- [setName](#) (value)

Setter for [name\(\)](#).
- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).

- `setEnabled (value)`
Setter for `enabled()`.
- `styleClass ()`
Custom css class(es).
- `setStyleClass (value)`
Setter for `styleClass()`.
- `style ()`
Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle (value)`
Setter for `style()`.
- `visibility ()`
If this widget is visible.
- `setVisibility (value)`
Setter for `visibility()`.
- `doStandaloneResizing ()`
If the widget handles to resize itself as needed.
- `setDoStandaloneResizing (value)`
Setter for `doStandaloneResizing()`.
- `mayFocus ()`
If the widget can have the keyboard focus.
- `setMayFocus (value)`
Setter for `mayFocus()`.
- `horizontalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity (value)`
Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity (value)`
Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing ()`
If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing (value)`
Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing ()`
If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing (value)`
Setter for `strictVerticalSizing()`.
- `vstretch ()`
Alias for `verticalStretchAffinity()`.
- `setVstretch (value)`
Setter for `vstretch()`.
- `hstretch ()`
Alias for `horizontalStretchAffinity()`.
- `setHstretch (value)`
Setter for `hstretch()`.
- `busy ()`
If the widget is in busy state, typically resulting in a loading animation.
- `setBusy (value)`
Setter for `busy()`.
- `registerBusy ()`

- Sets the widget to busy state and returns a token which helps returning to normal state.*
- [unregisterBusy](#) (token)
 - Drops a token and returns to normal state if no other tokens exist.*
- [hasFocus](#) ()
 - If the widget has the keyboard focus.*
- [innerSize](#) ()
 - Returns inner width and height of this widget.*
- [outerSize](#) ()
 - Returns outer width and height of this widget.*
- [OnDestroyed](#)
 - Triggered when this widget was removed somehow.*
- [OnResized](#)
 - Triggered when this widget was resized.*
- [OnVisibilityChanged](#)
 - Triggered when the visibility of this widget changed.*
- [OnPropertyChanged](#)
 - Triggered whenever a property of this widget changed its value.*
- [OnKeyDown](#)
 - Triggered when a keyboard key went down.*
- [OnKeyUp](#)
 - Triggered when a keyboard key came up.*
- [OnKeyPress](#)
 - Triggered when a keyboard key was pressed.*

1.68.1 Detailed Description

A box for password input from the user.

1.68.2 Member Function Documentation

1.68.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.68.2.2 autoCompleteFilter()

`autoCompleteFilter () [inherited]`

How to filter the autocomplete list.

Either 'startswith', 'contains', `function(itemtext, boxtext)` or undefined.

1.68.2.3 autoCompleteItems()

`autoCompleteItems () [inherited]`

A [clove.DataSource](#) with a list of autocomplete items to popup.

It is allowed to observe the `text` in order to generate live content for this list.

1.68.2.4 autoCompleteOpenForNoText()

`autoCompleteOpenForNoText () [inherited]`

If to show the autocomplete list when the edit box is empty.

1.68.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.68.2.6 busy()

`busy () [inherited]`

If the widget is in busy state, typically resulting in a loading animation.

1.68.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.68.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.68.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.68.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.68.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.68.2.12 `containingNameScope()`

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.68.2.13 `declareProperty()`

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.68.2.14 `doinit()`

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.68.2.15 `doinitEarly()`

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.68.2.16 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.68.2.17 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.68.2.18 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.68.2.19 effectiveVisibility()

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.68.2.20 enabled()

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.68.2.21 focus()

`focus () [inherited]`

Sets the focus to this widget.

1.68.2.22 getMinimalHeightForWidth()

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.68.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.68.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.68.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.68.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.68.2.27 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.68.2.28 hintText()

```
hintText ( ) [inherited]
```

The hint text.

Typically contains something like a label text. It is shown as a hint when the box is empty.

1.68.2.29 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.68.2.30 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.68.2.31 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.68.2.32 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.68.2.33 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.68.2.34 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class *class*.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.68.2.35 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.68.2.36 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.68.2.37 nameScope()

nameScope () [inherited]

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.68.2.38 OnChanged()

OnChanged [inherited]

Triggered when the text was changed.

This is a [clove.Event](#) instance. See Manual for details.

1.68.2.39 OnDestroyed()

OnDestroyed [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.68.2.40 OnKeyDown()

OnKeyDown [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.68.2.41 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.68.2.42 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.68.2.43 OnPopupTextSelected()

OnPopupTextSelected [inherited]

Triggered when a text was selected from the popup.

This is a [clove.Event](#) instance. See Manual for details.

1.68.2.44 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.68.2.45 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.68.2.46 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.68.2.47 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.68.2.48 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.68.2.49 readOnly()

```
readOnly ( ) [inherited]
```

If the edit box is read-only or editable by the user.

1.68.2.50 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.68.2.51 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.68.2.52 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.68.2.53 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.68.2.54 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.68.2.55 `setAutocompletionFilter()`

```
setAutocompletionFilter (
    value ) [inherited]
```

Setter for [autocompletionFilter\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.56 `setAutocompletionItems()`

```
setAutocompletionItems (
    value ) [inherited]
```

Setter for [autocompletionItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.57 `setAutocompletionOpenForNoText()`

```
setAutocompletionOpenForNoText (
    value ) [inherited]
```

Setter for [autocompletionOpenForNoText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.58 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.59 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.60 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.61 `setHintText()`

```
setHintText (
    value ) [inherited]
```

Setter for [hintText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.62 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.63 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.64 `setMayFocus()`

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.65 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.66 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.68.2.67 setReadOnly()

```
setReadOnly (
    value ) [inherited]
```

Setter for [readOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.68 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.69 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.70 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.71 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.72 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.68.2.73 `setText()`

```
setText (
    value ) [inherited]
```

Setter for [text\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.74 `setTextSelection()`

```
setTextSelection (
    ifrom,
    ito ) [inherited]
```

Selects the specified part of the text.

Parameters

<i>ifrom</i>	The selection begin index.
<i>ito</i>	The selection end index.

1.68.2.75 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.76 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.77 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.68.2.78 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.68.2.79 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.68.2.80 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.68.2.81 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.68.2.82 text()

```
text ( ) [inherited]
```

The current text.

1.68.2.83 textSelection()

```
textSelection ( ) [inherited]
```

Returns the current text selection indices.

1.68.2.84 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.68.2.85 `verticalStretchAffinity()`

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.68.2.86 `visibility()`

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.68.2.87 `vstretch()`

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

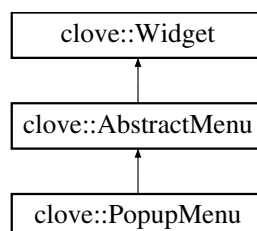
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.69 `clove::PopupMenu` Class Reference

A popup menu (i.e. a box with vertically listed actions).

Inheritance diagram for `clove::PopupMenu`:



Public Member Functions

- [expanderIndicatorDirection](#) ()
If the expander indicator is to be shown on the 'right' or 'left' side.
- [setExpanderIndicatorDirection](#) (value)
Setter for [expanderIndicatorDirection\(\)](#).
- static [show](#) (menu, showconfig)
Shows a popup menu.
- [actions](#) ()
The actions to be shown in this menu.
- [setActions](#) (value)
Setter for [actions\(\)](#).
- [OnActionTriggered](#)
Triggered when a menu action was chosen by the user for execution.
- [OnBeforeSubactionsExpanded](#)
Triggered just before a branch of subaction is expanded.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()

- Returns the preferred width in pixel this widget needs to have.*

 - [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
 - [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
 - [addStyleClass](#) (class)

Adds the css class `class` to this widget.
 - [removeStyleClass](#) (class)

Removes the css class `class` from this widget.
 - [isStyleClass](#) (class)

Checks if this widget contains the css class `class`.
 - [setStyleClassAssigned](#) (class, assigned)

Sets or unsets the css class `class` to this widget.
 - [containingNameScope](#) ()

The namespace this widget contains to.
 - [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
 - [remove](#) (removeconfig)

Removes this widget.
 - [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
 - [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
 - [childrenWidgets](#) ()

List of the children `clove::Widget` instances.
 - [parentWidget](#) ()

The parent `clove::Widget`.
 - [name](#) ()

The name of the widget.
 - [setName](#) (value)

Setter for [name\(\)](#).
 - [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
 - [setEnabled](#) (value)

Setter for [enabled\(\)](#).
 - [styleClass](#) ()

Custom css class(es).
 - [setStyleClass](#) (value)

Setter for [styleClass\(\)](#).
 - [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
 - [setStyle](#) (value)

Setter for [style\(\)](#).
 - [visibility](#) ()

If this widget is visible.
 - [setVisibility](#) (value)

Setter for [visibility\(\)](#).
 - [doStandaloneResizing](#) ()

If the widget handles to resize itself as needed.
 - [setDoStandaloneResizing](#) (value)

Setter for [doStandaloneResizing\(\)](#).

- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)

Triggered when a keyboard key went down.

- [OnKeyUp](#)

Triggered when a keyboard key came up.

- [OnKeyPress](#)

Triggered when a keyboard key was pressed.

1.69.1 Detailed Description

A popup menu (i.e. a box with vertically listed actions).

1.69.2 Member Function Documentation

1.69.2.1 actions()

```
actions ( ) [inherited]
```

The actions to be shown in this menu.

It is a list of action configuration objects, each having a structure like `{name:'myaction', label:'My menu action'}`. It can have a list of sub-items in the `subactions` property.

Instead of a simple list, it may also be a [clove::Datasource](#) with the action configuration structures aligned in rows.

Use [clove::MenuSeparator](#) for a separator.

One action allows this properties:

- `name`: The action name.
- `label`: The label string.
- `icon`: A [clove::Icon](#).
- `disabled`: If it is disabled.
- `invisible`: If it is invisible.
- `checkable`: If it is checkable.
- `checked`: If it is checked.

1.69.2.2 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.69.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.69.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.69.2.5 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.69.2.6 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.69.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.69.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.69.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.69.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.69.2.11 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.69.2.12 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.69.2.13 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.69.2.14 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.69.2.15 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.69.2.16 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.69.2.17 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.69.2.18 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.69.2.19 expanderIndicatorDirection()

```
expanderIndicatorDirection ( )
```

If the expander indicator is to be shown on the 'right' or 'left' side.

1.69.2.20 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.69.2.21 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.69.2.22 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.69.2.23 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.69.2.24 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.69.2.25 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.69.2.26 hasFocus()

`hasFocus () [inherited]`

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.69.2.27 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.69.2.28 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.69.2.29 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.69.2.30 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.69.2.31 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.69.2.32 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.69.2.33 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.69.2.34 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.69.2.35 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.69.2.36 OnActionTriggered()

`OnActionTriggered [inherited]`

Triggered when a menu action was chosen by the user for execution.

The event arguments contain the selected action name in `action`.

This is a [Clove.Event](#) instance. See Manual for details.

1.69.2.37 OnBeforeSubactionsExpanded()

`OnBeforeSubactionsExpanded [inherited]`

Triggered just before a branch of subaction is expanded.

Implement this event for populating it dynamically.

This is a [Clove.Event](#) instance. See Manual for details.

1.69.2.38 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [Clove.Event](#) instance. See Manual for details.

1.69.2.39 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [Clove.Event](#) instance. See Manual for details.

1.69.2.40 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [Clove.Event](#) instance. See Manual for details.

1.69.2.41 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.69.2.42 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.69.2.43 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.69.2.44 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.69.2.45 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.69.2.46 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.69.2.47 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.69.2.48 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.69.2.49 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.69.2.50 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.69.2.51 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.69.2.52 `setActions()`

```
setActions (
    value ) [inherited]
```

Setter for [actions\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.53 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.54 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.55 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.56 `setExpanderIndicatorDirection()`

```
setExpanderIndicatorDirection (
    value )
```

Setter for [expanderIndicatorDirection\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.57 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.58 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.59 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.60 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.61 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.69.2.62 `setStrictHorizontalSizing()`

```
setStrictHorizontalSizing (  
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.63 `setStrictVerticalSizing()`

```
setStrictVerticalSizing (  
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.64 `setStyle()`

```
setStyle (  
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.65 `setStyleClass()`

```
setStyleClass (  
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.66 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.69.2.67 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.68 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.69 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.69.2.70 show()

```
static show (
    menu,
    showconfig )
```

Shows a popup menu.

The root view of `menu` is expected to be a [Clove::PopupMenu](#) (or a subclass).

Parameters

<i>menu</i>	The widget configuration, as for Clove::build() , which specifies the popup menu.
<i>showconfig</i>	A configuration object which specifies some presentation aspects.

1.69.2.71 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.69.2.72 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.69.2.73 style()

`style () [inherited]`

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.69.2.74 styleClass()

`styleClass () [inherited]`

Custom css class(es).

1.69.2.75 unregisterBusy()

`unregisterBusy (
 token) [inherited]`

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.69.2.76 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.69.2.77 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.69.2.78 vstretch()

`vstretch ()` [inherited]

Alias for [verticalStretchAffinity\(\)](#).

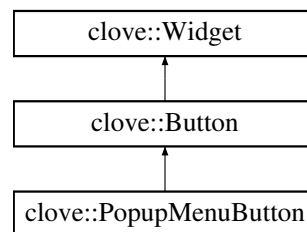
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.70 clove::PopupMenuButton Class Reference

A special [clove::Button](#) which opens a popup menu when it is triggered.

Inheritance diagram for `clove::PopupMenuButton`:



Public Member Functions

- [actions](#) ()
The menu actions. Same as for [clove::Menubar::actions\(\)](#).
- [setActions](#) (value)
Setter for [actions\(\)](#).
- [OnActionTriggered](#)
Triggered when a menu action was chosen by the user for execution.
- [label](#) ()
The label text.
- [setLabel](#) (value)
Setter for [label\(\)](#).
- [icon](#) ()
The optional [clove::Icon](#) button icon.
- [setIcon](#) (value)
Setter for [icon\(\)](#).
- [checkable](#) ()
If the button is checkable (i.e. has a checked-flag).
- [setCheckable](#) (value)
Setter for [checkable\(\)](#).
- [checked](#) ()
For a checkable button, return if it is checked.
- [setChecked](#) (value)
Setter for [checked\(\)](#).
- [OnClicked](#)

- Triggered when the user clicks on this button.*
- [declareProperty](#) (k, defaultV)
Declares a widget property.
 - [getProperty](#) (k)
General-purpose getter for widget properties.
 - [setProperty](#) (k, v)
General-purpose setter for widget properties.
 - [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
 - [init](#) (rootNameScope)
Initializes the widget.
 - [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
 - [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
 - [resize](#) ()
Applies the new widget size to internal content.
 - [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
 - [relayout](#) ()
Notifies the parent widget that a new geometry is required.
 - [focus](#) ()
Sets the focus to this widget.
 - [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
 - [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
 - [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
 - [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
 - [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
 - [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
 - [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
 - [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
 - [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
 - [addStyleClass](#) (css)
Adds the css class `css` to this widget.
 - [removeStyleClass](#) (css)
Removes the css class `css` from this widget.
 - [isStyleClass](#) (css)
Checks if this widget contains the css class `css`.
 - [setStyleClassAssigned](#) (css, assigned)
Sets or unsets the css class `css` to this widget.
 - [containingNameScope](#) ()
The namespace this widget contains to.

- `nameScope ()`
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- `remove (removeconfig)`
Removes this widget.
- `effectivelyEnabled ()`
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- `effectiveVisibility ()`
If this widget and all parent widgets are visible. See also `visibility()`.
- `childrenWidgets ()`
List of the children `clove::Widget` instances.
- `parentWidget ()`
The parent `clove::Widget`.
- `name ()`
The name of the widget.
- `setName (value)`
Setter for `name()`.
- `enabled ()`
If this widget is marked as enabled (i.e. can interact with the user).
- `setEnabled (value)`
Setter for `enabled()`.
- `styleClass ()`
Custom css class(es).
- `setStyleClass (value)`
Setter for `styleClass()`.
- `style ()`
Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle (value)`
Setter for `style()`.
- `visibility ()`
If this widget is visible.
- `setVisibility (value)`
Setter for `visibility()`.
- `doStandaloneResizing ()`
If the widget handles to resize itself as needed.
- `setDoStandaloneResizing (value)`
Setter for `doStandaloneResizing()`.
- `mayFocus ()`
If the widget can have the keyboard focus.
- `setMayFocus (value)`
Setter for `mayFocus()`.
- `horizontalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity (value)`
Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity (value)`
Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing ()`
If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing (value)`

- Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
 - If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
 - Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
 - Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
 - Setter for [vstretch\(\)](#).
- [hstretch](#) ()
 - Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 - Setter for [hstretch\(\)](#).
- [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 - Setter for [busy\(\)](#).
- [registerBusy](#) ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
 - Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
 - If the widget has the keyboard focus.
- [innerSize](#) ()
 - Returns inner width and height of this widget.
- [outerSize](#) ()
 - Returns outer width and height of this widget.
- [OnDestroyed](#)
 - Triggered when this widget was removed somehow.
- [OnResized](#)
 - Triggered when this widget was resized.
- [OnVisibilityChanged](#)
 - Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
 - Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
 - Triggered when a keyboard key went down.
- [OnKeyUp](#)
 - Triggered when a keyboard key came up.
- [OnKeyPress](#)
 - Triggered when a keyboard key was pressed.

1.70.1 Detailed Description

A special [clove::Button](#) which opens a popup menu when it is triggered.

1.70.2 Member Function Documentation

1.70.2.1 actions()

```
actions ( )
```

The menu actions. Same as for [clove::Menubar::actions\(\)](#).

1.70.2.2 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.70.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<code>k</code>	The widget property name.
<code>vb</code>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.70.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.70.2.5 checkable()

```
checkable ( ) [inherited]
```

If the button is checkable (i.e. has a checked-flag).

If it has, the checked-flag is controlled by [checked\(\)](#).

1.70.2.6 checked()

```
checked ( ) [inherited]
```

For a checkable button, return if it is checked.

See also [checkable\(\)](#).

User interactions do not toggle the checked flag by default. This must be scripted in own event handlers as needed.

1.70.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.70.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.70.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.70.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.70.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.70.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.70.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.70.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.70.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.70.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.70.2.17 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.70.2.18 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.70.2.19 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.70.2.20 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.70.2.21 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.70.2.22 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.70.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.70.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.70.2.25 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.70.2.26 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.70.2.27 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.70.2.28 `horizontalStretchAffinity()`

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.70.2.29 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.70.2.30 icon()

```
icon ( ) [inherited]
```

The optional [clove::Icon](#) button icon.

1.70.2.31 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.70.2.32 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.70.2.33 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.70.2.34 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.70.2.35 label()

```
label ( ) [inherited]
```

The label text.

1.70.2.36 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.70.2.37 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.70.2.38 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.70.2.39 OnActionTriggered()

`OnActionTriggered`

Triggered when a menu action was chosen by the user for execution.

The event arguments contain the selected action name in `action`.

This is a [clove.Event](#) instance. See Manual for details.

1.70.2.40 OnClicked()

`OnClicked` [inherited]

Triggered when the user clicks on this button.

This is a [clove.Event](#) instance. See Manual for details.

1.70.2.41 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.70.2.42 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.70.2.43 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.70.2.44 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.70.2.45 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.70.2.46 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.70.2.47 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.70.2.48 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.70.2.49 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.70.2.50 registerBusy()

`registerBusy ()` [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.70.2.51 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.70.2.52 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.70.2.53 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.70.2.54 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.70.2.55 setActions()

```
setActions (
    value )
```

Setter for [actions\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.56 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.57 setCheckable()

```
setCheckable (
    value ) [inherited]
```

Setter for [checkable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.58 setChecked()

```
setChecked (
    value ) [inherited]
```

Setter for [checked\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.59 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.60 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.61 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.62 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.63 setIcon()

```
setIcon (
    value ) [inherited]
```

Setter for [icon\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.64 setLabel()

```
setLabel (
    value ) [inherited]
```

Setter for [label\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.65 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.66 setName()

```
setName (  
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.67 setProperty()

```
setProperty (  
    k,  
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.70.2.68 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (  
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.69 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.70 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.71 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.72 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.70.2.73 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (  
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.74 `setVisibility()`

```
setVisibility (  
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.75 `setVstretch()`

```
setVstretch (  
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.70.2.76 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.70.2.77 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.70.2.78 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.70.2.79 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.70.2.80 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.70.2.81 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.70.2.82 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.70.2.83 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

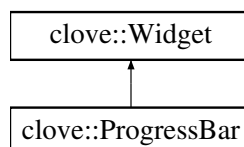
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.71 clove::ProgressBar Class Reference

A progress bar.

Inheritance diagram for `clove::ProgressBar`:



Public Member Functions

- [value](#) ()
The progress value between 0.0 and 1.0; or undefined for indeterminate progress.
- [setValue](#) (value)
Setter for [value\(\)](#).
- [orientation](#) ()
If to present the progress bar 'vertical'ly or 'horizontal'ly.
- [setOrientation](#) (value)
Setter for [orientation\(\)](#).
- [label](#) ()
An optional additional label text, which is displayed combined with the progress value.
- [setLabel](#) (value)
Setter for [label\(\)](#).
- [labelFunction](#) ()
A function(value, widget) which returns a custom label text.
- [setLabelFunction](#) (value)
Setter for [labelFunction\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()

- Returns the minimal width in pixel this widget needs to have.*

 - [getPreferredWidth](#) ()

Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
- [addClass](#) (class)

Adds the css class `class` to this widget.
- [removeClass](#) (class)

Removes the css class `class` from this widget.
- [hasClass](#) (class)

Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)

Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.
- [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)

Removes this widget.
- [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()

List of the children `clove::Widget` instances.
- [parentWidget](#) ()

The parent `clove::Widget`.
- [name](#) ()

The name of the widget.
- [setName](#) (value)

Setter for [name\(\)](#).
- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)

Setter for [enabled\(\)](#).
- [styleClass](#) ()

Custom css class(es).
- [setStyleClass](#) (value)

Setter for [styleClass\(\)](#).
- [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)

Setter for [style\(\)](#).
- [visibility](#) ()

If this widget is visible.
- [setVisibility](#) (value)

Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()

If the widget handles to resize itself as needed.

- [setDoStandaloneResizing \(value\)](#)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus \(\)](#)
If the widget can have the keyboard focus.
- [setMayFocus \(value\)](#)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity \(\)](#)
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity \(value\)](#)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity \(\)](#)
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity \(value\)](#)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing \(\)](#)
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing \(value\)](#)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing \(\)](#)
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing \(value\)](#)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch \(\)](#)
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch \(value\)](#)
Setter for [vstretch\(\)](#).
- [hstretch \(\)](#)
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch \(value\)](#)
Setter for [hstretch\(\)](#).
- [busy \(\)](#)
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy \(value\)](#)
Setter for [busy\(\)](#).
- [registerBusy \(\)](#)
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy \(token\)](#)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus \(\)](#)
If the widget has the keyboard focus.
- [innerSize \(\)](#)
Returns inner width and height of this widget.
- [outerSize \(\)](#)
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)

Triggered whenever a property of this widget changed its value.

- [OnKeyDown](#)

Triggered when a keyboard key went down.

- [OnKeyUp](#)

Triggered when a keyboard key came up.

- [OnKeyPress](#)

Triggered when a keyboard key was pressed.

1.71.1 Detailed Description

A progress bar.

1.71.2 Member Function Documentation

1.71.2.1 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.71.2.2 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<code>k</code>	The widget property name.
<code>vb</code>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.71.2.3 `busy()`

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.71.2.4 `childrenWidgets()`

```
childrenWidgets ( ) [inherited]
```

List of the children `clove::Widget` instances.

1.71.2.5 `computeMinimalHeightForWidth()`

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.71.2.6 `computeMinimalWidth()`

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.71.2.7 `computePreferredHeightForWidth()`

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.71.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.71.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.71.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.71.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.71.2.12 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.71.2.13 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.71.2.14 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.71.2.15 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.71.2.16 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.71.2.17 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.71.2.18 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.71.2.19 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.71.2.20 `getMinimalWidth()`

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.71.2.21 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.71.2.22 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.71.2.23 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.71.2.24 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.71.2.25 `horizontalStretchAffinity()`

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.71.2.26 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.71.2.27 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.71.2.28 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.71.2.29 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.71.2.30 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class *class*.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.71.2.31 label()

```
label ( )
```

An optional additional label text, which is displayed combined with the progress value.

1.71.2.32 labelFunction()

```
labelFunction ( )
```

A `function(value, widget)` which returns a custom label text.

1.71.2.33 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.71.2.34 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.71.2.35 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.71.2.36 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.71.2.37 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.71.2.38 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.71.2.39 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.71.2.40 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.71.2.41 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.71.2.42 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.71.2.43 orientation()

`orientation` ()

If to present the progress bar 'vertical'ly or 'horizontal'ly.

1.71.2.44 outerSize()

`outerSize` () [inherited]

Returns outer width and height of this widget.

1.71.2.45 parentWidget()

`parentWidget` () [inherited]

The parent [clove::Widget](#).

1.71.2.46 registerBusy()

`registerBusy` () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.71.2.47 relayout()

`relayout` () [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.71.2.48 remove()

`remove` (
 `removeconfig`) [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.71.2.49 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.71.2.50 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.71.2.51 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.52 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.53 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.54 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.55 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.56 setLabel()

```
setLabel (
    value )
```

Setter for [label\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.57 setLabelFunction()

```
setLabelFunction (
    value )
```

Setter for [labelFunction\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.58 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.59 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.60 setOrientation()

```
setOrientation (
    value )
```

Setter for [orientation\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.61 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.71.2.62 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.63 `setStrictVerticalSizing()`

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.64 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.65 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.66 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.71.2.67 setValue()

```
setValue (
    value )
```

Setter for [value\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.68 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.69 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.70 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.71.2.71 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.71.2.72 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.71.2.73 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.71.2.74 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.71.2.75 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.71.2.76 value()

value ()

The progress value between 0.0 and 1.0; or undefined for indeterminate progress.

1.71.2.77 verticalStretchAffinity()

verticalStretchAffinity () [inherited]

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.71.2.78 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.71.2.79 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

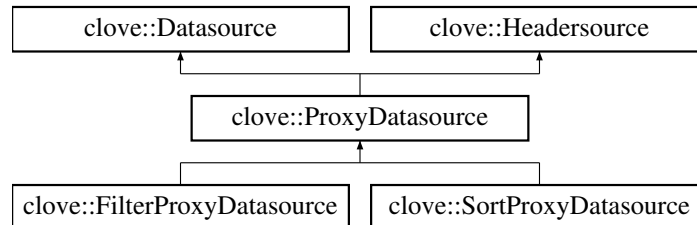
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.72 clove::ProxyDatasource Class Reference

A [clove::Datasource](#) implementation which proxies the content of another datasource in a somehow transformed way.

Inheritance diagram for `clove::ProxyDatasource`:



Public Member Functions

- [setDatasource](#) (datasource)
Sets the source datasource.
- [proxyPointerToNativePointer](#) (proyptr)
Translates a [clove::DatasourceValuePointer](#) from this proxy datasource to a one for the source datasource.
- [nativePointerToProxyPointer](#) (nativeptr)
Translates a [clove::DatasourceValuePointer](#) from the source datasource to a one for this proxy datasource.
- [refresh](#) ()
Refreshes the filtering.
- [getValue](#) (ptr)
Returns the value for a given node.
- [getMetadata](#) (ptr)
Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.
- [changeValue](#) (ptr, value)
Change the value for a given node.
- [rowCount](#) (parent)
Returns the number of rows for a given node.
- [columnCount](#) (parent)
Returns the number of columns for a given node.
- [valuePointer](#) (irow, icol, parent)
Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.
- [parent](#) (ptr)
Returns the parent node for a given node.
- [valuePointerNavigateInDepth](#) (ptr, direction, mayexpandfct)
Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.
- [OnDataInsert](#)
Triggered when a node insertion takes place.
- [OnDataRemove](#)
Triggered when a node removal takes place.
- [OnDataUpdate](#)
Triggered when a node data update takes place.
- [getRowHeader](#) (irow, parent)
Returns the header configuration for a given row.
- [getColumnHeader](#) (irow, parent)

- Returns the header configuration for a given column.*
- [rowHeadersVisible](#) (parent)
If row headers are visible.
- [columnHeadersVisible](#) (parent)
If column headers are visible.
- [OnHeaderDataInsert](#)
Triggered when a new row or column of header data was inserted.
- [OnHeaderDataRemove](#)
Triggered when a row or column of header data was removed.
- [OnHeaderDataUpdate](#)
Triggered when header data were updated.
- [OnHeaderVisibilityUpdated](#)
Triggered when header visibilities changed.

1.72.1 Detailed Description

A [clove::Datasource](#) implementation which proxies the content of another datasource in a somehow transformed way.

Note: This class is an abstract base class. Use one of the subclasses.

1.72.2 Member Function Documentation

1.72.2.1 `changeValue()`

```
changeValue (
    ptr,
    value ) [pure virtual], [inherited]
```

Change the value for a given node.

This method is called from external places, e.g. when a user makes changes in a datasource-connected widget.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>value</i>	The new value.

1.72.2.2 `columnCount()`

```
columnCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of columns for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.72.2.3 `columnHeadersVisible()`

```
columnHeadersVisible (  
    parent ) [pure virtual], [inherited]
```

If column headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.72.2.4 `getColumnHeader()`

```
getColumnHeader (  
    irow,  
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given column.

Parameters

<i>irow</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.72.2.5 `getMetadata()`

```
getMetadata (  
    ptr ) [pure virtual], [inherited]
```

Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.72.2.6 getRowHeader()

```
getRowHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given row.

Parameters

<i>irow</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.72.2.7 getValue()

```
getValue (
    ptr ) [pure virtual], [inherited]
```

Returns the value for a given node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.72.2.8 nativePointerToProxyPointer()

```
nativePointerToProxyPointer (
    nativeptr )
```

Translates a [clove::DatasourceValuePointer](#) from the source datasource to a one for this proxy datasource.

Parameters

<i>nativeptr</i>	A value pointer.
------------------	------------------

1.72.2.9 OnDataInsert()

```
OnDataInsert [inherited]
```

Triggered when a node insertion takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.72.2.10 OnDataRemove()

`OnDataRemove` [inherited]

Triggered when a node removal takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.72.2.11 OnDataUpdate()

`OnDataUpdate` [inherited]

Triggered when a node data update takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.72.2.12 OnHeaderDataInsert()

`OnHeaderDataInsert` [inherited]

Triggered when a new row or column of header data was inserted.

This is a [clove.Event](#) instance. See Manual for details.

1.72.2.13 OnHeaderDataRemove()

`OnHeaderDataRemove` [inherited]

Triggered when a row or column of header data was removed.

This is a [clove.Event](#) instance. See Manual for details.

1.72.2.14 OnHeaderDataUpdate()

`OnHeaderDataUpdate` [inherited]

Triggered when header data were updated.

This is a [clove.Event](#) instance. See Manual for details.

1.72.2.15 OnHeaderVisibilityUpdated()

`OnHeaderVisibilityUpdated` [inherited]

Triggered when header visibilities changed.

This is a [clove.Event](#) instance. See Manual for details.

1.72.2.16 parent()

`parent` (
 `ptr`) [pure virtual], [inherited]

Returns the parent node for a given node.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
------------	---

1.72.2.17 proxyPointerToNativePointer()

```
proxyPointerToNativePointer (
    proxyptr )
```

Translates a [clove::DatasourceValuePointer](#) from this proxy datasource to a one for the source datasource.

Parameters

<i>proxyptr</i>	A value pointer.
-----------------	------------------

1.72.2.18 refresh()

```
refresh ( )
```

Refreshes the filtering.

Call this when the outer conditions changed and the filters must be applied again.

1.72.2.19 rowCount()

```
rowCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of rows for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.72.2.20 rowHeadersVisible()

```
rowHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If row headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.72.2.21 `setDatasource()`

```
setDatasource (
    datasource )
```

Sets the source datasource.

Parameters

<i>datasource</i>	The source clove::Datasource .
-------------------	--

1.72.2.22 `valuePointer()`

```
valuePointer (
    irow,
    icol,
    parent ) [pure virtual], [inherited]
```

Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.

Parameters

<i>irow</i>	The row index.
<i>icol</i>	The column index.
<i>parent</i>	The parent as clove::DatasourceValuePointer .

1.72.2.23 `valuePointerNavigateInDepth()`

```
valuePointerNavigateInDepth (
    ptr,
    direction,
    mayexpandfct ) [inherited]
```

Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
<i>direction</i>	The direction (+1 or -1).
<i>mayexpandfct</i>	A function(<i>ptr</i>) which returns <code>true</code> iff this node's children are to be traversed.

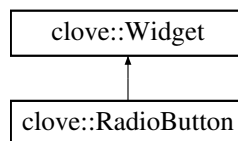
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.73 clove::RadioButton Class Reference

A radio button.

Inheritance diagram for clove::RadioButton:



Public Member Functions

- [label](#) ()
The label text.
- [setLabel](#) (value)
Setter for [label\(\)](#).
- [checked](#) ()
If this radio button is checked.
- [setChecked](#) (value)
Setter for [checked\(\)](#).
- [group](#) ()
The [clove::RadioGroup](#) this radio button belongs to.
- [setGroup](#) (value)
Setter for [group\(\)](#).
- [groupValue](#) ()
The currently selected value of the radiogroup this button belongs to. See [group\(\)](#).
- [setGroupValue](#) (value)
Setter for [groupValue\(\)](#).
- [valueDef](#) ()
The value this button is representing. You can use this together with [clove::RadioGroup::selectedValue\(\)](#).
- [setValueDef](#) (value)
Setter for [valueDef\(\)](#).
- [OnChanged](#)
Triggered when this radio button was selected.
- [OnClicked](#)
Triggered when this radio button was selected. Same as [OnChanged](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.

- `bindProperty` (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- `init` (rootNameScope)
Initializes the widget.
- `doinit` ()
Executes late widget initialization (i.e. after properties are applied).
- `doinitEarly` ()
Executes early widget initialization (i.e. before properties are applied).
- `resize` ()
Applies the new widget size to internal content.
- `doresize` ()
Corrects alignments of internal elements according to the new widget size.
- `relayout` ()
Notifies the parent widget that a new geometry is required.
- `focus` ()
Sets the focus to this widget.
- `isAlive` ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- `computeMinimalWidth` ()
Computes the minimal width in pixel this widget needs to have.
- `computePreferredWidth` ()
Computes the preferred width in pixel this widget needs to have.
- `computeMinimalHeightForWidth` (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- `computePreferredHeightForWidth` (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- `getMinimalWidth` ()
Returns the minimal width in pixel this widget needs to have.
- `getPreferredWidth` ()
Returns the preferred width in pixel this widget needs to have.
- `getMinimalHeightForWidth` (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- `getPreferredHeightForWidth` (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- `addStyleClass` (clss)
Adds the css class `clss` to this widget.
- `removeStyleClass` (clss)
Removes the css class `clss` from this widget.
- `isStyleClass` (clss)
Checks if this widget contains the css class `clss`.
- `setStyleClassAssigned` (clss, assigned)
Sets or unsets the css class `clss` to this widget.
- `containingNameScope` ()
The namespace this widget contains to.
- `nameScope` ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- `remove` (removeconfig)
Removes this widget.
- `effectivelyEnabled` ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- `effectiveVisibility` ()

- If this widget and all parent widgets are visible. See also [visibility\(\)](#).*

 - [childrenWidgets](#) ()

List of the children [clove::Widget](#) instances.
 - [parentWidget](#) ()

The parent [clove::Widget](#).
 - [name](#) ()

The name of the widget.
 - [setName](#) (value)

Setter for [name\(\)](#).
 - [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
 - [setEnabled](#) (value)

Setter for [enabled\(\)](#).
 - [styleClass](#) ()

Custom css class(es).
 - [setStyleClass](#) (value)

Setter for [styleClass\(\)](#).
 - [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
 - [setStyle](#) (value)

Setter for [style\(\)](#).
 - [visibility](#) ()

If this widget is visible.
 - [setVisibility](#) (value)

Setter for [visibility\(\)](#).
 - [doStandaloneResizing](#) ()

If the widget handles to resize itself as needed.
 - [setDoStandaloneResizing](#) (value)

Setter for [doStandaloneResizing\(\)](#).
 - [mayFocus](#) ()

If the widget can have the keyboard focus.
 - [setMayFocus](#) (value)

Setter for [mayFocus\(\)](#).
 - [horizontalStretchAffinity](#) ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
 - [setHorizontalStretchAffinity](#) (value)

Setter for [horizontalStretchAffinity\(\)](#).
 - [verticalStretchAffinity](#) ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
 - [setVerticalStretchAffinity](#) (value)

Setter for [verticalStretchAffinity\(\)](#).
 - [strictHorizontalSizing](#) ()

If the widget is strictly forbidden to get additional horizontal space.
 - [setStrictHorizontalSizing](#) (value)

Setter for [strictHorizontalSizing\(\)](#).
 - [strictVerticalSizing](#) ()

If the widget is strictly forbidden to get additional vertical space.
 - [setStrictVerticalSizing](#) (value)

Setter for [strictVerticalSizing\(\)](#).
 - [vstretch](#) ()

Alias for [verticalStretchAffinity\(\)](#).

- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.73.1 Detailed Description

A radio button.

Typically some radio buttons are assigned to one [clove::RadioGroup](#) so the user can choose one of them (while a [clove::CheckBox](#) allows 0..n choices).

1.73.2 Member Function Documentation

1.73.2.1 [addStyleClass\(\)](#)

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.73.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.73.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.73.2.4 checked()

```
checked ( )
```

If this radio button is checked.

See also [clove::RadioGroup::selected\(\)](#).

1.73.2.5 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.73.2.6 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.73.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.73.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.73.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.73.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.73.2.11 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.73.2.12 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.73.2.13 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.73.2.14 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.73.2.15 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.73.2.16 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.73.2.17 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.73.2.18 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.73.2.19 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.73.2.20 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.73.2.21 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.73.2.22 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.73.2.23 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.73.2.24 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.73.2.25 `group()`

```
group ( )
```

The [clove::RadioGroup](#) this radio button belongs to.

Note: It is possible to assign strings to it, see [clove::RadioGroup::getGroupByPublicName\(\)](#).

1.73.2.26 `groupValue()`

```
groupValue ( )
```

The currently selected value of the radiogroup this button belongs to. See [group\(\)](#).

1.73.2.27 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.73.2.28 `horizontalStretchAffinity()`

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.73.2.29 `hstretch()`

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.73.2.30 `init()`

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.73.2.31 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.73.2.32 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.73.2.33 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.73.2.34 label()

```
label ( )
```

The label text.

1.73.2.35 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.73.2.36 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.73.2.37 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.73.2.38 OnChanged()

`OnChanged`

Triggered when this radio button was selected.

This is a [clove.Event](#) instance. See Manual for details.

1.73.2.39 OnClicked()

`OnClicked`

Triggered when this radio button was selected. Same as OnChanged.

This is a [clove.Event](#) instance. See Manual for details.

1.73.2.40 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.73.2.41 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.73.2.42 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.73.2.43 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.73.2.44 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.73.2.45 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.73.2.46 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.73.2.47 `outerSize()`

```
outerSize ( ) [inherited]
```

Returns outer width and height of this widget.

1.73.2.48 `parentWidget()`

```
parentWidget ( ) [inherited]
```

The parent [Clove::Widget](#).

1.73.2.49 `registerBusy()`

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.73.2.50 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.73.2.51 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.73.2.52 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.73.2.53 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.73.2.54 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.55 setChecked()

```
setChecked (
    value )
```

Setter for [checked\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.56 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.57 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.58 setGroup()

```
setGroup (
    value )
```

Setter for [group\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.59 setGroupValue()

```
setGroupValue (
    value )
```

Setter for [groupValue\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.60 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.61 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.62 setLabel()

```
setLabel (
    value )
```

Setter for [label\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.63 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.64 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.65 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.73.2.66 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (  
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.67 setStrictVerticalSizing()

```
setStrictVerticalSizing (  
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.68 setStyle()

```
setStyle (  
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.69 setStyleClass()

```
setStyleClass (  
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.70 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.73.2.71 `setValueDef()`

```
setValueDef (
    value )
```

Setter for [valueDef\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.72 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.73 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.74 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.73.2.75 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.73.2.76 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.73.2.77 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.73.2.78 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.73.2.79 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.73.2.80 valueDef()

```
valueDef ( )
```

The value this button is representing. You can use this together with [clove::RadioGroup::selectedValue\(\)](#).

1.73.2.81 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.73.2.82 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.73.2.83 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.74 clove::RadioGroup Class Reference

Groups some [clove::RadioButton](#) together in a logical way, so the user can select exactly one of them.

Public Member Functions

- [RadioGroup](#) ()
- [selected](#) ()
Returns the [clove::RadioButton](#) which is currently selected in this group.
- [select](#) (w)
Selects a button in the group.
- [OnChanged](#)
Triggered when another button was selected in this group.
- static [getGroupByPublicName](#) (name)
Returns a [clove::RadioGroup](#) by name. Either returns an already existing one, or creates a new one.
- [selectedIndex](#) ()
Returns the position index of the currently selected child in this group.
- [selectedValue](#) ()
Returns the value of the currently selected child in this group.
- [selectByIndex](#) (i)
Selects a child by its position index.
- [selectByValue](#) (v)
Selects a child by its value. It will also understand position indexes as fallback.
- [unselectAll](#) ()
Unselects all childs in this group.

1.74.1 Detailed Description

Groups some [clove::RadioButton](#) together in a logical way, so the user can select exactly one of them.

Assign buttons to a group by setting [clove::RadioButton::group\(\)](#).

1.74.2 Constructor & Destructor Documentation

1.74.2.1 RadioGroup()

`RadioGroup ()`

1.74.3 Member Function Documentation

1.74.3.1 getGroupByPublicName()

```
static getGroupByPublicName (
    name )
```

Returns a `clove::RadioGroup` by name. Either returns an already existing one, or creates a new one.

Developers of library functionality should not use this function due to risks of naming clashes.

Parameters

<i>name</i>	The group name.
-------------	-----------------

1.74.3.2 OnChanged()

`OnChanged`

Triggered when another button was selected in this group.

This is a `clove.Event` instance. See Manual for details.

1.74.3.3 select()

```
select (
    w )
```

Selects a button in the group.

Parameters

<i>w</i>	The <code>clove::RadioButton</code> to select.
----------	--

1.74.3.4 selectByIndex()

```
selectByIndex (
```

`i`)

Selects a child by its position index.

Parameters

<code>i</code>	The position index.
----------------	---------------------

1.74.3.5 `selectByValue()`

`selectByValue` (
 `v`)

Selects a child by its value. It will also understand position indexes as fallback.

Parameters

<code>v</code>	The value.
----------------	------------

1.74.3.6 `selected()`

`selected` ()

Returns the `clove::RadioButton` which is currently selected in this group.

1.74.3.7 `selectedIndex()`

`selectedIndex` ()

Returns the position index of the currently selected child in this group.

1.74.3.8 `selectedValue()`

`selectedValue` ()

Returns the value of the currently selected child in this group.

1.74.3.9 unselectAll()

```
unselectAll ( )
```

Unselects all childs in this group.

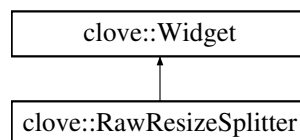
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.75 clove::RawResizeSplitter Class Reference

The actual splitter in a [clove::ResizeSplitter](#).

Inheritance diagram for `clove::RawResizeSplitter`:



Public Member Functions

- [OnMoveBegin](#)
Triggered when the user begins to move the splitter.
- [OnMove](#)
Triggered subsequently while the user moves the splitter.
- [OnMoveEnd](#)
Triggered when the user ends moving the splitter.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()

- Notifies the parent widget that a new geometry is required.*

 - [focus](#) ()

Sets the focus to this widget.
 - [isAlive](#) ()

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
 - [computeMinimalWidth](#) ()

Computes the minimal width in pixel this widget needs to have.
 - [computePreferredWidth](#) ()

Computes the preferred width in pixel this widget needs to have.
 - [computeMinimalHeightForWidth](#) (w)

Computes the minimal height in pixel this widget needs to have for a given width.
 - [computePreferredHeightForWidth](#) (w)

Computes the preferred height in pixel this widget needs to have for a given width.
 - [getMinimalWidth](#) ()

Returns the minimal width in pixel this widget needs to have.
 - [getPreferredWidth](#) ()

Returns the preferred width in pixel this widget needs to have.
 - [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
 - [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
 - [addStyleClass](#) (clss)

Adds the css class `clss` to this widget.
 - [removeStyleClass](#) (clss)

Removes the css class `clss` from this widget.
 - [isStyleClass](#) (clss)

Checks if this widget contains the css class `clss`.
 - [setStyleClassAssigned](#) (clss, assigned)

Sets or unsets the css class `clss` to this widget.
 - [containingNameScope](#) ()

The namespace this widget contains to.
 - [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
 - [remove](#) (removeconfig)

Removes this widget.
 - [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
 - [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
 - [childrenWidgets](#) ()

List of the children `clove::Widget` instances.
 - [parentWidget](#) ()

The parent `clove::Widget`.
 - [name](#) ()

The name of the widget.
 - [setName](#) (value)

Setter for [name\(\)](#).
 - [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
 - [setEnabled](#) (value)

Setter for [enabled\(\)](#).

- `styleClass ()`
Custom css class(es).
- `setStyleClass (value)`
Setter for `styleClass()`.
- `style ()`
Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle (value)`
Setter for `style()`.
- `visibility ()`
If this widget is visible.
- `setVisibility (value)`
Setter for `visibility()`.
- `doStandaloneResizing ()`
If the widget handles to resize itself as needed.
- `setDoStandaloneResizing (value)`
Setter for `doStandaloneResizing()`.
- `mayFocus ()`
If the widget can have the keyboard focus.
- `setMayFocus (value)`
Setter for `mayFocus()`.
- `horizontalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity (value)`
Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity (value)`
Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing ()`
If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing (value)`
Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing ()`
If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing (value)`
Setter for `strictVerticalSizing()`.
- `vstretch ()`
Alias for `verticalStretchAffinity()`.
- `setVstretch (value)`
Setter for `vstretch()`.
- `hstretch ()`
Alias for `horizontalStretchAffinity()`.
- `setHstretch (value)`
Setter for `hstretch()`.
- `busy ()`
If the widget is in busy state, typically resulting in a loading animation.
- `setBusy (value)`
Setter for `busy()`.
- `registerBusy ()`
Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy (token)`

- Drops a token and returns to normal state if no other tokens exist.*
- [hasFocus \(\)](#)
If the widget has the keyboard focus.
- [innerSize \(\)](#)
Returns inner width and height of this widget.
- [outerSize \(\)](#)
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.75.1 Detailed Description

The actual splitter in a [clove::ResizeSplitter](#).

Not intended for direct usage in a user interface!

1.75.2 Member Function Documentation

1.75.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.75.2.2 bindProperty()

```
bindProperty (
```

```

    k,
    vb ) [inherited]

```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.75.2.3 busy()

```

busy ( ) [inherited]

```

If the widget is in busy state, typically resulting in a loading animation.

1.75.2.4 childrenWidgets()

```

childrenWidgets ( ) [inherited]

```

List of the children [clove::Widget](#) instances.

1.75.2.5 computeMinimalHeightForWidth()

```

computeMinimalHeightForWidth (
    w ) [inherited]

```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.75.2.6 computeMinimalWidth()

```

computeMinimalWidth ( ) [inherited]

```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.75.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.75.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.75.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.75.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.75.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.75.2.12 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.75.2.13 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.75.2.14 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.75.2.15 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.75.2.16 effectiveVisibility()

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.75.2.17 enabled()

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.75.2.18 focus()

`focus () [inherited]`

Sets the focus to this widget.

1.75.2.19 getMinimalHeightForWidth()

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.75.2.20 getMinimalWidth()

`getMinimalWidth () [inherited]`

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.75.2.21 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.75.2.22 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.75.2.23 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.75.2.24 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.75.2.25 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.75.2.26 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.75.2.27 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.75.2.28 innerSize()

`innerSize () [inherited]`

Returns inner width and height of this widget.

1.75.2.29 isAlive()

`isAlive () [inherited]`

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.75.2.30 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.75.2.31 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.75.2.32 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.75.2.33 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.75.2.34 OnDestroyed()

```
OnDestroyed [inherited]
```

Triggered when this widget was removed somehow.

This is a [Clove.Event](#) instance. See Manual for details.

1.75.2.35 OnKeyDown()

OnKeyDown [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.75.2.36 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.75.2.37 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.75.2.38 OnMove()

OnMove

Triggered subsequently while the user moves the splitter.

This is a [clove.Event](#) instance. See Manual for details.

1.75.2.39 OnMoveBegin()

OnMoveBegin

Triggered when the user begins to move the splitter.

This is a [clove.Event](#) instance. See Manual for details.

1.75.2.40 OnMoveEnd()

OnMoveEnd

Triggered when the user ends moving the splitter.

This is a [clove.Event](#) instance. See Manual for details.

1.75.2.41 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.75.2.42 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.75.2.43 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.75.2.44 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.75.2.45 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.75.2.46 registerBusy()

`registerBusy ()` [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.75.2.47 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.75.2.48 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.75.2.49 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.75.2.50 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.75.2.51 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.52 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.53 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.54 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.55 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.56 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.57 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.58 `setProperty()`

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.75.2.59 `setStrictHorizontalSizing()`

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.60 `setStrictVerticalSizing()`

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.61 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.62 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.63 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.75.2.64 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.65 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.66 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.75.2.67 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretch←Affinity\(\)](#)==0.

1.75.2.68 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretch←Affinity\(\)](#)==0.

1.75.2.69 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.75.2.70 styleClass()

styleClass () [inherited]

Custom css class(es).

1.75.2.71 unregisterBusy()

unregisterBusy (
 token) [inherited]

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.75.2.72 verticalStretchAffinity()

verticalStretchAffinity () [inherited]

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.75.2.73 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.75.2.74 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

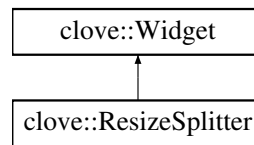
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.76 clove::ResizeSplitter Class Reference

A container for multiple widgets with splitters for manual resizing between them.

Inheritance diagram for clove::ResizeSplitter:



Public Member Functions

- [orientation](#) ()
If the two widgets are stacked 'vertical'ly or 'horizontal'ly.
- [setOrientation](#) (value)
Setter for [orientation\(\)](#).
- [body](#) ()
The list of body widgets (i.e. the widgets to show separated by splitters) as widget configurations like for [clove::build\(\)](#).
- [setBody](#) (value)
Setter for [body\(\)](#).
- [bodyWidgets](#) ()
Returns the list of all body widgets as [clove::Widget](#) instances.
- [setBodyWidget](#) (i, config)
Sets (overrides) the body widget at a given position.
- [addBodyWidget](#) (config)
Adds a new body widget to the end.
- [insertBodyWidget](#) (i, config)
Inserts a new body widget somewhere.
- [sizeFractions](#) ()
The sizes of each widget inside this resize splitter as list of numbers. It has a positive entry for each widget, describing the amount of room it takes. The values have just relative meaning (and are typically fractions of 1).
- [setSizeFractions](#) (value)
Setter for [sizeFractions\(\)](#).
- [showOnly](#) ()
Specifies if to show just one of the two sides (0, 1, undefined).
- [setShowOnly](#) (value)
Setter for [showOnly\(\)](#).
- [splitterWidth](#) ()
The splitter width (height for vertical orientation) as css length.
- [setSplitterWidth](#) (value)
Setter for [splitterWidth\(\)](#).
- [splitterVisibility](#) ()
The visibility of the splitter. See [clove::Widget::visibility\(\)](#) for details.
- [setSplitterVisibility](#) (value)
Setter for [splitterVisibility\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)

- General-purpose getter for widget properties.*

 - [setProperty](#) (k, v)

General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)

Initializes the widget.
- [doinit](#) ()

Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()

Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()

Applies the new widget size to internal content.
- [doresize](#) ()

Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()

Notifies the parent widget that a new geometry is required.
- [focus](#) ()

Sets the focus to this widget.
- [isAlive](#) ()

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()

Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()

Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)

Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)

Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()

Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()

Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)

Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)

Removes the css class `css` from this widget.
- [isStyleClass](#) (css)

Checks if this widget contains the css class `css`.
- [setStyleClassAssigned](#) (css, assigned)

Sets or unsets the css class `css` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.
- [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)

Removes this widget.

- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children [clove::Widget](#) instances.
- [parentWidget](#) ()
The parent [clove::Widget](#).
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)

- [vstretch](#) ()
 Setter for [strictVerticalSizing\(\)](#).
- [setVstretch](#) (value)
 Alias for [verticalStretchAffinity\(\)](#).
- [hstretch](#) ()
 Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 Setter for [hstretch\(\)](#).
- [busy](#) ()
 If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 Setter for [busy\(\)](#).
- [registerBusy](#) ()
 Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
 Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
 If the widget has the keyboard focus.
- [innerSize](#) ()
 Returns inner width and height of this widget.
- [outerSize](#) ()
 Returns outer width and height of this widget.
- [OnDestroyed](#)
 Triggered when this widget was removed somehow.
- [OnResized](#)
 Triggered when this widget was resized.
- [OnVisibilityChanged](#)
 Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
 Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
 Triggered when a keyboard key went down.
- [OnKeyUp](#)
 Triggered when a keyboard key came up.
- [OnKeyPress](#)
 Triggered when a keyboard key was pressed.

1.76.1 Detailed Description

A container for multiple widgets with splitters for manual resizing between them.

1.76.2 Member Function Documentation

1.76.2.1 addBodyWidget()

```
addBodyWidget (
    config )
```

Adds a new body widget to the end.

Parameters

<i>config</i>	The widget as widget configurations like for clove::build() .
---------------	---

1.76.2.2 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.76.2.3 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.76.2.4 `body()`

```
body ( )
```

The list of body widgets (i.e. the widgets to show separated by splitters) as widget configurations like for [clove↵::build\(\)](#).

1.76.2.5 `bodyWidgets()`

```
bodyWidgets ( )
```

Returns the list of all body widgets as [clove::Widget](#) instances.

1.76.2.6 `busy()`

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.76.2.7 `childrenWidgets()`

```
childrenWidgets ( ) [inherited]
```

List of the children `clove::Widget` instances.

1.76.2.8 `computeMinimalHeightForWidth()`

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.76.2.9 `computeMinimalWidth()`

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.76.2.10 `computePreferredHeightForWidth()`

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.76.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.76.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.76.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.76.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.76.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.76.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.76.2.17 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.76.2.18 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.76.2.19 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.76.2.20 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.76.2.21 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.76.2.22 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.76.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.76.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.76.2.25 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.76.2.26 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.76.2.27 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.76.2.28 `horizontalStretchAffinity()`

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.76.2.29 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.76.2.30 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.76.2.31 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.76.2.32 insertBodyWidget()

```
insertBodyWidget (
    i,
    config )
```

Inserts a new body widget somewhere.

Parameters

<i>i</i>	The widget position.
<i>config</i>	The widget as widget configurations like for clove::build() .

1.76.2.33 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.76.2.34 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.76.2.35 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.76.2.36 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.76.2.37 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.76.2.38 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.76.2.39 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.76.2.40 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.76.2.41 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.76.2.42 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.76.2.43 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.76.2.44 `OnVisibilityChanged()`

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.76.2.45 `orientation()`

`orientation` ()

If the two widgets are stacked `'vertical'`ly or `'horizontal'`ly.

1.76.2.46 `outerSize()`

`outerSize` () [inherited]

Returns outer width and height of this widget.

1.76.2.47 `parentWidget()`

`parentWidget` () [inherited]

The parent [clove::Widget](#).

1.76.2.48 `registerBusy()`

`registerBusy` () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.76.2.49 `relayout()`

`relayout` () [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.76.2.50 `remove()`

`remove` (
 `removeconfig`) [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.76.2.51 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.76.2.52 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.76.2.53 setBody()

```
setBody (
    value )
```

Setter for [body\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.54 setBodyWidget()

```
setBodyWidget (
    i,
    config )
```

Sets (overrides) the body widget at a given position.

Parameters

<i>i</i>	The widget position.
<i>config</i>	The widget as widget configurations like for clove::build() .

1.76.2.55 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.56 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.57 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.58 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (  
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.59 setHstretch()

```
setHstretch (  
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.60 setMayFocus()

```
setMayFocus (  
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.61 setName()

```
setName (  
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.62 setOrientation()

```
setOrientation (
    value )
```

Setter for [orientation\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.63 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.76.2.64 setShowOnly()

```
setShowOnly (
    value )
```

Setter for [showOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.65 setSizeFractions()

```
setSizeFractions (
    value )
```

Setter for [sizeFractions\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.66 setSplitterVisibility()

```
setSplitterVisibility (
    value )
```

Setter for [splitterVisibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.67 setSplitterWidth()

```
setSplitterWidth (
    value )
```

Setter for [splitterWidth\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.68 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.69 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.70 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.71 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.72 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.76.2.73 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.74 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.75 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.76.2.76 `showOnly()`

```
showOnly ( )
```

Specifies if to show just one of the two sides (0, 1, undefined).

1.76.2.77 `sizeFractions()`

```
sizeFractions ( )
```

The sizes of each widget inside this resize splitter as list of numbers. It has a positive entry for each widget, describing the amount of room it takes. The values have just relative meaning (and are typically fractions of 1).

1.76.2.78 `splitterVisibility()`

```
splitterVisibility ( )
```

The visibility of the splitter. See [clove::Widget::visibility\(\)](#) for details.

1.76.2.79 `splitterWidth()`

```
splitterWidth ( )
```

The splitter width (height for vertical orientation) as css length.

1.76.2.80 `strictHorizontalSizing()`

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.76.2.81 strictVerticalSizing()

`strictVerticalSizing () [inherited]`

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.76.2.82 style()

`style () [inherited]`

Custom css style string. You should not use that, but `styleClass()` instead.

1.76.2.83 styleClass()

`styleClass () [inherited]`

Custom css class(es).

1.76.2.84 unregisterBusy()

`unregisterBusy (
 token) [inherited]`

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.76.2.85 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.76.2.86 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.76.2.87 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

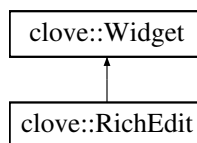
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.77 clove::RichEdit Class Reference

A user input box for text with rich formatting.

Inheritance diagram for `clove::RichEdit`:



Public Member Functions

- [htmlContent](#) ()
The content text as html string.
- [setHtmlContent](#) (value)
Setter for [htmlContent\(\)](#).
- [richeditbox](#) ()
Returns the inner [clove::RichEditBox](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)

- Initializes the widget.*
- [doinit \(\)](#)
 - Executes late widget initialization (i.e. after properties are applied).*
- [doinitEarly \(\)](#)
 - Executes early widget initialization (i.e. before properties are applied).*
- [resize \(\)](#)
 - Applies the new widget size to internal content.*
- [doresize \(\)](#)
 - Corrects alignments of internal elements according to the new widget size.*
- [relayout \(\)](#)
 - Notifies the parent widget that a new geometry is required.*
- [focus \(\)](#)
 - Sets the focus to this widget.*
- [isAlive \(\)](#)
 - Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).*
- [computeMinimalWidth \(\)](#)
 - Computes the minimal width in pixel this widget needs to have.*
- [computePreferredWidth \(\)](#)
 - Computes the preferred width in pixel this widget needs to have.*
- [computeMinimalHeightForWidth \(w\)](#)
 - Computes the minimal height in pixel this widget needs to have for a given width.*
- [computePreferredHeightForWidth \(w\)](#)
 - Computes the preferred height in pixel this widget needs to have for a given width.*
- [getMinimalWidth \(\)](#)
 - Returns the minimal width in pixel this widget needs to have.*
- [getPreferredWidth \(\)](#)
 - Returns the preferred width in pixel this widget needs to have.*
- [getMinimalHeightForWidth \(w\)](#)
 - Returns the minimal height in pixel this widget needs to have for a given width.*
- [getPreferredHeightForWidth \(w\)](#)
 - Returns the preferred height in pixel this widget needs to have for a given width.*
- [addStyleClass \(css\)](#)
 - Adds the css class `css` to this widget.*
- [removeStyleClass \(css\)](#)
 - Removes the css class `css` from this widget.*
- [isStyleClass \(css\)](#)
 - Checks if this widget contains the css class `css`.*
- [setStyleClassAssigned \(css, assigned\)](#)
 - Sets or unsets the css class `css` to this widget.*
- [containingNameScope \(\)](#)
 - The namespace this widget contains to.*
- [nameScope \(\)](#)
 - The own namespace (or the namespace it contains to, if this widget does not bring a new one).*
- [remove \(removeconfig\)](#)
 - Removes this widget.*
- [effectivelyEnabled \(\)](#)
 - If this widget is enabled and not marked as busy (i.e. can interact with the user).*
- [effectiveVisibility \(\)](#)
 - If this widget and all parent widgets are visible. See also [visibility\(\)](#).*
- [childrenWidgets \(\)](#)
 - List of the children [clove::Widget](#) instances.*

- `parentWidget ()`
The parent `clove::Widget`.
- `name ()`
The name of the widget.
- `setName (value)`
Setter for `name()`.
- `enabled ()`
If this widget is marked as enabled (i.e. can interact with the user).
- `setEnabled (value)`
Setter for `enabled()`.
- `styleClass ()`
Custom css class(es).
- `setStyleClass (value)`
Setter for `styleClass()`.
- `style ()`
Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle (value)`
Setter for `style()`.
- `visibility ()`
If this widget is visible.
- `setVisibility (value)`
Setter for `visibility()`.
- `doStandaloneResizing ()`
If the widget handles to resize itself as needed.
- `setDoStandaloneResizing (value)`
Setter for `doStandaloneResizing()`.
- `mayFocus ()`
If the widget can have the keyboard focus.
- `setMayFocus (value)`
Setter for `mayFocus()`.
- `horizontalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity (value)`
Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity ()`
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity (value)`
Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing ()`
If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing (value)`
Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing ()`
If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing (value)`
Setter for `strictVerticalSizing()`.
- `vstretch ()`
Alias for `verticalStretchAffinity()`.
- `setVstretch (value)`
Setter for `vstretch()`.
- `hstretch ()`

- Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 - Setter for [hstretch\(\)](#).
- [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 - Setter for [busy\(\)](#).
- [registerBusy](#) ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
 - Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
 - If the widget has the keyboard focus.
- [innerSize](#) ()
 - Returns inner width and height of this widget.
- [outerSize](#) ()
 - Returns outer width and height of this widget.
- [OnDestroyed](#)
 - Triggered when this widget was removed somehow.
- [OnResized](#)
 - Triggered when this widget was resized.
- [OnVisibilityChanged](#)
 - Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
 - Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
 - Triggered when a keyboard key went down.
- [OnKeyUp](#)
 - Triggered when a keyboard key came up.
- [OnKeyPress](#)
 - Triggered when a keyboard key was pressed.

1.77.1 Detailed Description

A user input box for text with rich formatting.

This widget includes formatting toolbars. For a bare scriptable edit box, see [clove::RichEditBox](#).

1.77.2 Member Function Documentation

1.77.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.77.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.77.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.77.2.4 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.77.2.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.77.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.77.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.77.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.77.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.77.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.77.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.77.2.12 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.77.2.13 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.77.2.14 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.77.2.15 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.77.2.16 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.77.2.17 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.77.2.18 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.77.2.19 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.77.2.20 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.77.2.21 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.77.2.22 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.77.2.23 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.77.2.24 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.77.2.25 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.77.2.26 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.77.2.27 htmlContent()

```
htmlContent ( )
```

The content text as html string.

1.77.2.28 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.77.2.29 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.77.2.30 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.77.2.31 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.77.2.32 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.77.2.33 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.77.2.34 nameScope()

nameScope () [inherited]

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.77.2.35 OnDestroyed()

OnDestroyed [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.77.2.36 OnKeyDown()

OnKeyDown [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.77.2.37 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.77.2.38 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.77.2.39 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.77.2.40 OnResized()

```
OnResized [inherited]
```

Triggered when this widget was resized.

This is a [Clove.Event](#) instance. See Manual for details.

1.77.2.41 OnVisibilityChanged()

```
OnVisibilityChanged [inherited]
```

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [Clove.Event](#) instance. See Manual for details.

1.77.2.42 outerSize()

```
outerSize ( ) [inherited]
```

Returns outer width and height of this widget.

1.77.2.43 parentWidget()

```
parentWidget ( ) [inherited]
```

The parent [Clove::Widget](#).

1.77.2.44 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.77.2.45 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.77.2.46 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.77.2.47 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.77.2.48 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.77.2.49 richeditbox()

```
richeditbox ( )
```

Returns the inner [clove::RichEditBox](#).

1.77.2.50 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.51 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.52 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.53 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.54 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.55 setHtmlContent()

```
setHtmlContent (
    value )
```

Setter for [htmlContent\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.56 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.57 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.58 `setProperty()`

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.77.2.59 `setStrictHorizontalSizing()`

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.60 `setStrictVerticalSizing()`

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.61 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.62 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.63 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.77.2.64 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.65 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.66 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.77.2.67 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretch←Affinity\(\)](#)==0.

1.77.2.68 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretch←Affinity\(\)](#)==0.

1.77.2.69 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.77.2.70 styleClass()

styleClass () [inherited]

Custom css class(es).

1.77.2.71 unregisterBusy()

unregisterBusy (
 token) [inherited]

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.77.2.72 verticalStretchAffinity()

verticalStretchAffinity () [inherited]

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.77.2.73 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.77.2.74 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

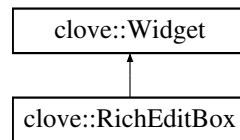
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.78 clove::RichEditBox Class Reference

A user input box for text with rich formatting.

Inheritance diagram for clove::RichEditBox:



Public Member Functions

- [htmlContent](#) ()
The content text as html string.
- [setHtmlContent](#) (value)
Setter for [htmlContent\(\)](#).
- [selection](#) ()
The current selection (as a web-api Range object).
- [setSelection](#) (value)
Setter for [selection\(\)](#).
- [contentHtmlNode](#) ()
Returns the html content dom node.
- [toggleSelectionBold](#) ()
Toggles the bold-flag for the current selection.
- [toggleSelectionItalic](#) ()
Toggles the italic-flag for the current selection.
- [toggleSelectionUnderline](#) ()
Toggles the underline-flag for the current selection.
- [selectionIncreaseFontSize](#) ()
Increases the font size for the current selection.
- [selectionDecreaseFontSize](#) ()
Decreases the font size for the current selection.
- [selectionSetForegroundColor](#) (c)
Sets the foreground color for the current selection.
- [selectionSetBackgroundColor](#) (c)
Sets the background color for the current selection.
- [selectionInsertList](#) (config)
Inserts a list at the current place.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.

- [doinit \(\)](#)
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly \(\)](#)
Executes early widget initialization (i.e. before properties are applied).
- [resize \(\)](#)
Applies the new widget size to internal content.
- [doresize \(\)](#)
Corrects alignments of internal elements according to the new widget size.
- [relayout \(\)](#)
Notifies the parent widget that a new geometry is required.
- [focus \(\)](#)
Sets the focus to this widget.
- [isAlive \(\)](#)
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth \(\)](#)
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth \(\)](#)
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth \(w\)](#)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth \(w\)](#)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth \(\)](#)
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth \(\)](#)
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth \(w\)](#)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth \(w\)](#)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass \(css\)](#)
Adds the css class `css` to this widget.
- [removeStyleClass \(css\)](#)
Removes the css class `css` from this widget.
- [isStyleClass \(css\)](#)
Checks if this widget contains the css class `css`.
- [setStyleClassAssigned \(css, assigned\)](#)
Sets or unsets the css class `css` to this widget.
- [containingNameScope \(\)](#)
The namespace this widget contains to.
- [nameScope \(\)](#)
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove \(removeconfig\)](#)
Removes this widget.
- [effectivelyEnabled \(\)](#)
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility \(\)](#)
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets \(\)](#)
List of the children [clove::Widget](#) instances.
- [parentWidget \(\)](#)

- The parent `clove::Widget`.
- `name` ()
 - The name of the widget.
- `setName` (value)
 - Setter for `name()`.
- `enabled` ()
 - If this widget is marked as enabled (i.e. can interact with the user).
- `setEnabled` (value)
 - Setter for `enabled()`.
- `styleClass` ()
 - Custom css class(es).
- `setStyleClass` (value)
 - Setter for `styleClass()`.
- `style` ()
 - Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle` (value)
 - Setter for `style()`.
- `visibility` ()
 - If this widget is visible.
- `setVisibility` (value)
 - Setter for `visibility()`.
- `doStandaloneResizing` ()
 - If the widget handles to resize itself as needed.
- `setDoStandaloneResizing` (value)
 - Setter for `doStandaloneResizing()`.
- `mayFocus` ()
 - If the widget can have the keyboard focus.
- `setMayFocus` (value)
 - Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity` (value)
 - Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity` ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity` (value)
 - Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing` ()
 - If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing` (value)
 - Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing` ()
 - If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing` (value)
 - Setter for `strictVerticalSizing()`.
- `vstretch` ()
 - Alias for `verticalStretchAffinity()`.
- `setVstretch` (value)
 - Setter for `vstretch()`.
- `hstretch` ()
 - Alias for `horizontalStretchAffinity()`.

- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.78.1 Detailed Description

A user input box for text with rich formatting.

This widget allows to be controlled from external toolbars. For a widget including a formatting toolbar, see [clove::RichEdit](#).

1.78.2 Member Function Documentation

1.78.2.1 [addStyleClass\(\)](#)

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.78.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.78.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.78.2.4 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.78.2.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.78.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.78.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.78.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.78.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.78.2.10 `contentHtmlNode()`

```
contentHtmlNode ( )
```

Returns the html content dom node.

1.78.2.11 `declareProperty()`

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.78.2.12 `doinit()`

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.78.2.13 `doinitEarly()`

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.78.2.14 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.78.2.15 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.78.2.16 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.78.2.17 effectiveVisibility()

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.78.2.18 enabled()

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.78.2.19 focus()

`focus () [inherited]`

Sets the focus to this widget.

1.78.2.20 getMinimalHeightForWidth()

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.78.2.21 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.78.2.22 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.78.2.23 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.78.2.24 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.78.2.25 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.78.2.26 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.78.2.27 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.78.2.28 htmlContent()

```
htmlContent ( )
```

The content text as html string.

1.78.2.29 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.78.2.30 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.78.2.31 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.78.2.32 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class *class*.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.78.2.33 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.78.2.34 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.78.2.35 nameScope()

nameScope () [inherited]

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.78.2.36 OnDestroyed()

OnDestroyed [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.78.2.37 OnKeyDown()

OnKeyDown [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.78.2.38 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.78.2.39 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.78.2.40 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.78.2.41 OnResized()

```
OnResized [inherited]
```

Triggered when this widget was resized.

This is a [Clove.Event](#) instance. See Manual for details.

1.78.2.42 OnVisibilityChanged()

```
OnVisibilityChanged [inherited]
```

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [Clove.Event](#) instance. See Manual for details.

1.78.2.43 outerSize()

```
outerSize ( ) [inherited]
```

Returns outer width and height of this widget.

1.78.2.44 parentWidget()

```
parentWidget ( ) [inherited]
```

The parent [Clove::Widget](#).

1.78.2.45 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.78.2.46 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.78.2.47 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.78.2.48 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.78.2.49 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.78.2.50 selection()

```
selection ( )
```

The current selection (as a web-api Range object).

1.78.2.51 selectionDecreaseFontSize()

```
selectionDecreaseFontSize ( )
```

Decreases the font size for the current selection.

1.78.2.52 selectionIncreaseFontSize()

```
selectionIncreaseFontSize ( )
```

Increases the font size for the current selection.

1.78.2.53 selectionInsertList()

```
selectionInsertList (
    config )
```

Inserts a list at the current place.

config may contain:

- `ordered`: If the list is ordered.

Parameters

<code>config</code>	The list configuration.
---------------------	-------------------------

1.78.2.54 selectionSetBackgroundColor()

```
selectionSetBackgroundColor (
    c )
```

Sets the background color for the current selection.

Parameters

<code>c</code>	The color string.
----------------	-------------------

1.78.2.55 selectionSetForegroundColor()

```
selectionSetForegroundColor (
    c )
```

Sets the foreground color for the current selection.

Parameters

<code>c</code>	The color string.
----------------	-------------------

1.78.2.56 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.57 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.58 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.59 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.60 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.61 setHtmlContent()

```
setHtmlContent (
    value )
```

Setter for [htmlContent\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.62 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.63 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.64 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.78.2.65 setSelection()

```
setSelection (
    value )
```

Setter for [selection\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.66 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.67 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.68 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.69 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.70 setStyleClassAssigned()

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.78.2.71 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.72 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.73 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.78.2.74 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.78.2.75 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.78.2.76 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.78.2.77 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.78.2.78 toggleSelectionBold()

```
toggleSelectionBold ( )
```

Toggles the bold-flag for the current selection.

1.78.2.79 toggleSelectionItalic()

```
toggleSelectionItalic ( )
```

Toggles the italic-flag for the current selection.

1.78.2.80 toggleSelectionUnderline()

```
toggleSelectionUnderline ( )
```

Toggles the underline-flag for the current selection.

1.78.2.81 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.78.2.82 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.78.2.83 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.78.2.84 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

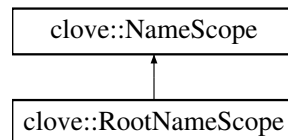
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.79 clove::RootNameScope Class Reference

A standalone namespace.

Inheritance diagram for clove::RootNameScope:



Public Member Functions

- [getByName](#) (name)
Finds a widget by name within this namespace.
- [addChildNameScope](#) (childnamespace)
Adds a namespace to the children of this one, so a searcher will also search in this child.

1.79.1 Detailed Description

A standalone namespace.

1.79.2 Member Function Documentation

1.79.2.1 addChildNameScope()

```
addChildNameScope (
    childnamespace ) [inherited]
```

Adds a namespace to the children of this one, so a searcher will also search in this child.

Parameters

<i>childnamespace</i>	The child namespace.
-----------------------	----------------------

1.79.2.2 getByName()

```
getByName (
    name ) [inherited]
```

Finds a widget by name within this namespace.

Parameters

<i>name</i>	The widget name.
-------------	------------------

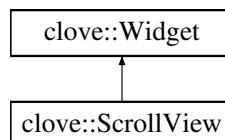
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.80 clove::ScrollView Class Reference

A container for making the children scrollable.

Inheritance diagram for clove::ScrollView:



Public Member Functions

- [body](#) ()
The inner widget as widget configuration like for [clove::build\(\)](#).
- [setBody](#) (value)
Setter for [body\(\)](#).
- [bodyWidget](#) ()
Returns the body as [clove::Widget](#).
- [bodySize](#) ()
The (outer) size of the body widget.
- [verticalScrollPosition](#) ()
The vertical scroll position.
- [setVerticalScrollPosition](#) (value)

- Setter for [verticalScrollPosition\(\)](#).
- [horizontalScrollPosition](#) ()
 - The horizontal scroll position.
- [setHorizontalScrollPosition](#) (value)
 - Setter for [horizontalScrollPosition\(\)](#).
- [declareProperty](#) (k, defaultV)
 - Declares a widget property.
- [getProperty](#) (k)
 - General-purpose getter for widget properties.
- [setProperty](#) (k, v)
 - General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
 - Binds a [DataBinding](#) to a property. Read [Manual](#) for details about data bindings.
- [init](#) (rootNameScope)
 - Initializes the widget.
- [doinit](#) ()
 - Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
 - Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
 - Applies the new widget size to internal content.
- [doresize](#) ()
 - Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
 - Notifies the parent widget that a new geometry is required.
- [focus](#) ()
 - Sets the focus to this widget.
- [isAlive](#) ()
 - Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
 - Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
 - Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
 - Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
 - Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
 - Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
 - Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
 - Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
 - Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)
 - Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)
 - Removes the css class `css` from this widget.
- [isStyleClass](#) (css)
 - Checks if this widget contains the css class `css`.

- [setStyleClassAssigned](#) (clss, assigned)
Sets or unsets the css class `clss` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children [clove::Widget](#) instances.
- [parentWidget](#) ()
The parent [clove::Widget](#).
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)

- Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
 - If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
 - Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
 - If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
 - Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
 - Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
 - Setter for [vstretch\(\)](#).
- [hstretch](#) ()
 - Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 - Setter for [hstretch\(\)](#).
- [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 - Setter for [busy\(\)](#).
- [registerBusy](#) ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
 - Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
 - If the widget has the keyboard focus.
- [innerSize](#) ()
 - Returns inner width and height of this widget.
- [outerSize](#) ()
 - Returns outer width and height of this widget.
- [OnDestroyed](#)
 - Triggered when this widget was removed somehow.
- [OnResized](#)
 - Triggered when this widget was resized.
- [OnVisibilityChanged](#)
 - Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
 - Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
 - Triggered when a keyboard key went down.
- [OnKeyUp](#)
 - Triggered when a keyboard key came up.
- [OnKeyPress](#)
 - Triggered when a keyboard key was pressed.

1.80.1 Detailed Description

A container for making the children scrollable.

1.80.2 Member Function Documentation

1.80.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.80.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.80.2.3 body()

```
body ( )
```

The inner widget as widget configuration like for [clove::build\(\)](#).

1.80.2.4 bodySize()

```
bodySize ( )
```

The (outer) size of the body widget.

1.80.2.5 bodyWidget()

```
bodyWidget ( )
```

Returns the body as [clove::Widget](#).

1.80.2.6 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.80.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.80.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.80.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.80.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.80.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.80.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.80.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.80.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.80.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.80.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.80.2.17 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.80.2.18 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.80.2.19 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.80.2.20 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.80.2.21 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.80.2.22 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.80.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.80.2.24 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.80.2.25 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.80.2.26 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.80.2.27 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.80.2.28 horizontalScrollPosition()

```
horizontalScrollPosition ( )
```

The horizontal scroll position.

1.80.2.29 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.80.2.30 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.80.2.31 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.80.2.32 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.80.2.33 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.80.2.34 `isStyleClass()`

```
isStyleClass (
    css ) [inherited]
```

Checks if this widget contains the css class `css`.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.80.2.35 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.80.2.36 `name()`

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.80.2.37 `nameScope()`

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.80.2.38 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.80.2.39 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.80.2.40 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.80.2.41 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.80.2.42 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.80.2.43 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.80.2.44 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.80.2.45 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.80.2.46 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.80.2.47 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.80.2.48 relayout()

relayout () [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.80.2.49 remove()

remove (
 removeconfig) [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.80.2.50 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.80.2.51 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.80.2.52 setBody()

```
setBody (
    value )
```

Setter for [body\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.53 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.54 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.55 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.56 setHorizontalScrollPosition()

```
setHorizontalScrollPosition (
    value )
```

Setter for [horizontalScrollPosition\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.57 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.58 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.59 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.60 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.61 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.80.2.62 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.63 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.64 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for `style()`.

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.65 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for `styleClass()`.

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.66 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.80.2.67 `setVerticalScrollPosition()`

```
setVerticalScrollPosition (
    value )
```

Setter for [verticalScrollPosition\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.68 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (  
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.69 setVisibility()

```
setVisibility (  
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.70 setVstretch()

```
setVstretch (  
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.80.2.71 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.80.2.72 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.80.2.73 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.80.2.74 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.80.2.75 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.80.2.76 verticalScrollPosition()

`verticalScrollPosition ()`

The vertical scroll position.

1.80.2.77 verticalStretchAffinity()

`verticalStretchAffinity ()` [inherited]

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.80.2.78 visibility()

`visibility ()` [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.80.2.79 vstretch()

`vstretch ()` [inherited]

Alias for [verticalStretchAffinity\(\)](#).

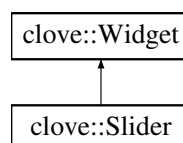
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.81 clove::Slider Class Reference

A slider allows the user to choose a number from a given value interval.

Inheritance diagram for `clove::Slider`:



Public Member Functions

- [min](#) ()
The minimum value.
- [setMin](#) (value)
Setter for [min\(\)](#).
- [max](#) ()
The maximum value.
- [setMax](#) (value)
Setter for [max\(\)](#).
- [stepsize](#) ()
The step size.
- [setStepsize](#) (value)
Setter for [stepsize\(\)](#).
- [value](#) ()
The current slider value.
- [setValue](#) (value)
Setter for [value\(\)](#).
- [orientation](#) ()
If to present the slider 'vertical'ly or 'horizontal'ly.
- [setOrientation](#) (value)
Setter for [orientation\(\)](#).
- [OnChanged](#)
Triggered when the slider value changed.
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()

- Computes the preferred width in pixel this widget needs to have.*

 - [computeMinimalHeightForWidth](#) (w)

Computes the minimal height in pixel this widget needs to have for a given width.

 - [computePreferredHeightForWidth](#) (w)
- Computes the preferred height in pixel this widget needs to have for a given width.*
- [getMinimalWidth](#) ()
- Returns the minimal width in pixel this widget needs to have.*
- [getPreferredWidth](#) ()
- Returns the preferred width in pixel this widget needs to have.*
- [getMinimalHeightForWidth](#) (w)
- Returns the minimal height in pixel this widget needs to have for a given width.*
- [getPreferredHeightForWidth](#) (w)
- Returns the preferred height in pixel this widget needs to have for a given width.*
- [addStyleClass](#) (class)
- Adds the css class `class` to this widget.*
- [removeStyleClass](#) (class)
- Removes the css class `class` from this widget.*
- [isStyleClass](#) (class)
- Checks if this widget contains the css class `class`.*
- [setStyleClassAssigned](#) (class, assigned)
- Sets or unsets the css class `class` to this widget.*
- [containingNameScope](#) ()
- The namespace this widget contains to.*
- [nameScope](#) ()
- The own namespace (or the namespace it contains to, if this widget does not bring a new one).*
- [remove](#) (removeconfig)
- Removes this widget.*
- [effectivelyEnabled](#) ()
- If this widget is enabled and not marked as busy (i.e. can interact with the user).*
- [effectiveVisibility](#) ()
- If this widget and all parent widgets are visible. See also [visibility\(\)](#).*
- [childrenWidgets](#) ()
- List of the children `clove::Widget` instances.*
- [parentWidget](#) ()
- The parent `clove::Widget`.*
- [name](#) ()
- The name of the widget.*
- [setName](#) (value)
- Setter for [name\(\)](#).*
- [enabled](#) ()
- If this widget is marked as enabled (i.e. can interact with the user).*
- [setEnabled](#) (value)
- Setter for [enabled\(\)](#).*
- [styleClass](#) ()
- Custom css class(es).*
- [setStyleClass](#) (value)
- Setter for [styleClass\(\)](#).*
- [style](#) ()
- Custom css style string. You should not use that, but [styleClass\(\)](#) instead.*
- [setStyle](#) (value)
- Setter for [style\(\)](#).*

- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility](#)().
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing](#)().
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus](#)().
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity](#)().
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity](#)().
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing](#)().
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing](#)().
- [vstretch](#) ()
Alias for [verticalStretchAffinity](#)().
- [setVstretch](#) (value)
Setter for [vstretch](#)().
- [hstretch](#) ()
Alias for [horizontalStretchAffinity](#)().
- [setHstretch](#) (value)
Setter for [hstretch](#)().
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy](#)().
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)

Triggered when this widget was removed somehow.

- [OnResized](#)

Triggered when this widget was resized.

- [OnVisibilityChanged](#)

Triggered when the visibility of this widget changed.

- [OnPropertyChanged](#)

Triggered whenever a property of this widget changed its value.

- [OnKeyDown](#)

Triggered when a keyboard key went down.

- [OnKeyUp](#)

Triggered when a keyboard key came up.

- [OnKeyPress](#)

Triggered when a keyboard key was pressed.

1.81.1 Detailed Description

A slider allows the user to choose a number from a given value interval.

The user chooses a value by moving a handle alongside a line.

1.81.2 Member Function Documentation

1.81.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.81.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The Clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.81.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.81.2.4 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [Clove::Widget](#) instances.

1.81.2.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.81.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.81.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.81.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.81.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.81.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.81.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.81.2.12 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.81.2.13 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.81.2.14 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.81.2.15 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.81.2.16 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.81.2.17 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.81.2.18 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.81.2.19 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.81.2.20 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.81.2.21 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.81.2.22 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.81.2.23 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.81.2.24 `hasFocus()`

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.81.2.25 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.81.2.26 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.81.2.27 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.81.2.28 innerSize()

`innerSize () [inherited]`

Returns inner width and height of this widget.

1.81.2.29 isAlive()

`isAlive () [inherited]`

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.81.2.30 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.81.2.31 max()

```
max ( )
```

The maximum value.

1.81.2.32 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.81.2.33 min()

```
min ( )
```

The minimum value.

1.81.2.34 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.81.2.35 nameScope()

nameScope () [inherited]

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.81.2.36 OnChanged()

OnChanged

Triggered when the slider value changed.

This is a [clove.Event](#) instance. See Manual for details.

1.81.2.37 OnDestroyed()

OnDestroyed [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.81.2.38 OnKeyDown()

OnKeyDown [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.81.2.39 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.81.2.40 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.81.2.41 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.81.2.42 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.81.2.43 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.81.2.44 orientation()

`orientation` ()

If to present the slider 'vertical'ly or 'horizontal'ly.

1.81.2.45 outerSize()

`outerSize` () [inherited]

Returns outer width and height of this widget.

1.81.2.46 parentWidget()

`parentWidget` () [inherited]

The parent [clove::Widget](#).

1.81.2.47 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.81.2.48 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.81.2.49 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.81.2.50 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.81.2.51 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.81.2.52 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.53 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.54 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.55 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (  
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.56 setHstretch()

```
setHstretch (  
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.57 setMax()

```
setMax (  
    value )
```

Setter for [max\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.58 setMayFocus()

```
setMayFocus (  
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.59 setMin()

```
setMin (
    value )
```

Setter for [min\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.60 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.61 setOrientation()

```
setOrientation (
    value )
```

Setter for [orientation\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.62 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.81.2.63 setStepsize()

```
setStepsize (
    value )
```

Setter for [stepsize\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.64 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.65 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.66 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.67 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.68 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.81.2.69 setValue()

```
setValue (
    value )
```

Setter for [value\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.70 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.71 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.72 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.81.2.73 `stepsize()`

```
stepsize ( )
```

The step size.

1.81.2.74 `strictHorizontalSizing()`

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.81.2.75 `strictVerticalSizing()`

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.81.2.76 `style()`

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.81.2.77 `styleClass()`

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.81.2.78 `unregisterBusy()`

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.81.2.79 value()

```
value ( )
```

The current slider value.

1.81.2.80 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.81.2.81 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.81.2.82 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

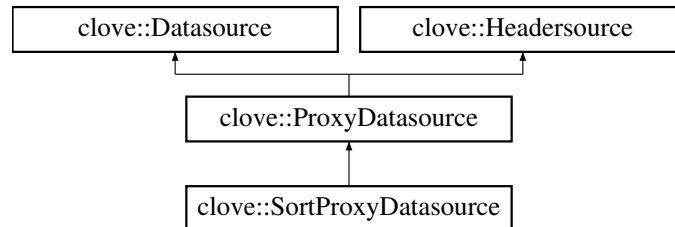
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.82 clove::SortProxyDatasource Class Reference

A [clove::Datasource](#) implementation which sorts another [clove::Datasource](#).

Inheritance diagram for `clove::SortProxyDatasource`:



Public Member Functions

- [SortProxyDatasource](#) (datasource, rowcomparefct, colcomparefct)
- [setRowComparator](#) (rowcomparefct)
Sets the row comparator function.
- [setColumnComparator](#) (colcomparefct)
Sets the column comparator function.
- [setDatasource](#) (datasource)
Sets the source datasource.
- [proxyPointerToNativePointer](#) (proxyptr)
Translates a [clove::DatasourceValuePointer](#) from this proxy datasource to a one for the source datasource.
- [nativePointerToProxyPointer](#) (nativeptr)
Translates a [clove::DatasourceValuePointer](#) from the source datasource to a one for this proxy datasource.
- [refresh](#) ()
Refreshes the filtering.
- [getValue](#) (ptr)
Returns the value for a given node.
- [getMetadata](#) (ptr)
Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.
- [changeValue](#) (ptr, value)
Change the value for a given node.
- [rowCount](#) (parent)
Returns the number of rows for a given node.
- [columnCount](#) (parent)
Returns the number of columns for a given node.
- [valuePointer](#) (irow, icol, parent)
Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.
- [parent](#) (ptr)
Returns the parent node for a given node.
- [valuePointerNavigateInDepth](#) (ptr, direction, mayexpandfct)
Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.
- [OnDataInsert](#)
Triggered when a node insertion takes place.
- [OnDataRemove](#)
Triggered when a node removal takes place.
- [OnDataUpdate](#)

- Triggered when a node data update takes place.*
- [getRowHeader](#) (*irow*, *parent*)
Returns the header configuration for a given row.
- [getColumnHeader](#) (*irow*, *parent*)
Returns the header configuration for a given column.
- [rowHeadersVisible](#) (*parent*)
If row headers are visible.
- [columnHeadersVisible](#) (*parent*)
If column headers are visible.
- [OnHeaderDataInsert](#)
Triggered when a new row or column of header data was inserted.
- [OnHeaderDataRemove](#)
Triggered when a row or column of header data was removed.
- [OnHeaderDataUpdate](#)
Triggered when header data were updated.
- [OnHeaderVisibilityUpdated](#)
Triggered when header visibilities changed.

1.82.1 Detailed Description

A [clove::Datasource](#) implementation which sorts another [clove::Datasource](#).

1.82.2 Constructor & Destructor Documentation

1.82.2.1 SortProxyDatasource()

```
SortProxyDatasource (
    datasource,
    rowcomparefct,
    colcomparefct )
```

Parameters

<i>datasource</i>	The source clove::Datasource .
<i>rowcomparefct</i>	A function(<i>datasource</i> , <i>irow</i> , <i>jrow</i> , <i>parent</i>) returning a negative value if <i>irow</i> points to a row lower than <i>jrow</i> , a positive value if it is greater, or 0 if both are equal.
<i>colcomparefct</i>	A function(<i>datasource</i> , <i>icol</i> , <i>jcol</i> , <i>parent</i>) returning a negative value if <i>icol</i> points to a column lower than <i>jcol</i> , a positive value if it is greater, or 0 if both are equal.

1.82.3 Member Function Documentation

1.82.3.1 `changeValue()`

```
changeValue (
    ptr,
    value ) [pure virtual], [inherited]
```

Change the value for a given node.

This method is called from external places, e.g. when a user makes changes in a datasource-connected widget.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>value</i>	The new value.

1.82.3.2 `columnCount()`

```
columnCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of columns for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.82.3.3 `columnHeadersVisible()`

```
columnHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If column headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.82.3.4 `getColumnHeader()`

```
getColumnHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given column.

Parameters

<i>irow</i>	The column index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.82.3.5 `getMetadata()`

```
getMetadata (
    ptr ) [pure virtual], [inherited]
```

Returns an object of metadata information (like icons, status infos used for styling, ...). Optional.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.82.3.6 `getRowHeader()`

```
getRowHeader (
    irow,
    parent ) [pure virtual], [inherited]
```

Returns the header configuration for a given row.

Parameters

<i>irow</i>	The row index.
<i>parent</i>	The parent node as clove::DatasourceValuePointer .

1.82.3.7 `getValue()`

```
getValue (
    ptr ) [pure virtual], [inherited]
```

Returns the value for a given node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.82.3.8 nativePointerToProxyPointer()

```
nativePointerToProxyPointer (
    nativeptr ) [inherited]
```

Translates a [clove::DatasourceValuePointer](#) from the source datasource to a one for this proxy datasource.

Parameters

<i>nativeptr</i>	A value pointer.
------------------	------------------

1.82.3.9 OnDataInsert()

```
OnDataInsert [inherited]
```

Triggered when a node insertion takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.82.3.10 OnDataRemove()

```
OnDataRemove [inherited]
```

Triggered when a node removal takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.82.3.11 OnDataUpdate()

```
OnDataUpdate [inherited]
```

Triggered when a node data update takes place.

This is a [clove.Event](#) instance. See Manual for details.

1.82.3.12 OnHeaderDataInsert()

```
OnHeaderDataInsert [inherited]
```

Triggered when a new row or column of header data was inserted.

This is a [clove.Event](#) instance. See Manual for details.

1.82.3.13 OnHeaderDataRemove()

```
OnHeaderDataRemove [inherited]
```

Triggered when a row or column of header data was removed.

This is a [clove.Event](#) instance. See Manual for details.

1.82.3.14 OnHeaderDataUpdate()

```
OnHeaderDataUpdate [inherited]
```

Triggered when header data were updated.

This is a [clove.Event](#) instance. See Manual for details.

1.82.3.15 OnHeaderVisibilityUpdated()

```
OnHeaderVisibilityUpdated [inherited]
```

Triggered when header visibilities changed.

This is a [clove.Event](#) instance. See Manual for details.

1.82.3.16 parent()

```
parent (
    ptr ) [pure virtual], [inherited]
```

Returns the parent node for a given node.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
------------	---

1.82.3.17 proxyPointerToNativePointer()

```
proxyPointerToNativePointer (
    proxyptr ) [inherited]
```

Translates a [clove::DatasourceValuePointer](#) from this proxy datasource to a one for the source datasource.

Parameters

<i>proxyptr</i>	A value pointer.
-----------------	------------------

1.82.3.18 refresh()

```
refresh ( ) [inherited]
```

Refreshes the filtering.

Call this when the outer conditions changed and the filters must be applied again.

1.82.3.19 rowCount()

```
rowCount (
    parent ) [pure virtual], [inherited]
```

Returns the number of rows for a given node.

Parameters

<i>parent</i>	The parent as clove::DatasourceValuePointer .
---------------	---

1.82.3.20 rowHeadersVisible()

```
rowHeadersVisible (
    parent ) [pure virtual], [inherited]
```

If row headers are visible.

Parameters

<i>parent</i>	The parent node as clove::DatasourceValuePointer .
---------------	--

1.82.3.21 setColumnComparator()

```
setColumnComparator (
    colcomparefct )
```

Sets the column comparator function.

Parameters

<i>colcomparefct</i>	The column comparator function. See constructor for details.
----------------------	--

1.82.3.22 setDatasource()

```
setDatasource (
    datasource ) [inherited]
```

Sets the source datasource.

Parameters

<i>datasource</i>	The source clove::Datasource .
-------------------	--

1.82.3.23 setRowComparator()

```
setRowComparator (
    rowcomparefct )
```

Sets the row comparator function.

Parameters

<i>rowcomparefct</i>	The row comparator function. See constructor for details.
----------------------	---

1.82.3.24 valuePointer()

```
valuePointer (
    irow,
    icol,
    parent ) [pure virtual], [inherited]
```

Constructs and returns a [clove::DatasourceValuePointer](#) for a given node.

Parameters

<i>irow</i>	The row index.
<i>icol</i>	The column index.
<i>parent</i>	The parent as clove::DatasourceValuePointer .

1.82.3.25 valuePointerNavigateInDepth()

```
valuePointerNavigateInDepth (
```

```
ptr,
direction,
mayexpandfct ) [inherited]
```

Returns a [clove::DatasourceValuePointer](#) resulting in the original one by navigation in depth.

Parameters

<i>ptr</i>	A node as clove::DatasourceValuePointer .
<i>direction</i>	The direction (+1 or -1).
<i>mayexpandfct</i>	A function(<i>ptr</i>) which returns <code>true</code> iff this node's children are to be traversed.

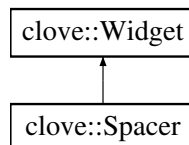
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.83 clove::Spacer Class Reference

A spacer widget for filling gaps in a layout.

Inheritance diagram for `clove::Spacer`:



Public Member Functions

- [declareProperty](#) (*k*, *defaultV*)
Declares a widget property.
- [getProperty](#) (*k*)
General-purpose getter for widget properties.
- [setProperty](#) (*k*, *v*)
General-purpose setter for widget properties.
- [bindProperty](#) (*k*, *vb*)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (*rootNameScope*)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.

- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addClass](#) (class)
Adds the css class `class` to this widget.
- [removeClass](#) (class)
Removes the css class `class` from this widget.
- [isStyleClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)

- Setter for [enabled\(\)](#).
- [styleClass](#) ()
 - Custom css class(es).
- [setStyleClass](#) (value)
 - Setter for [styleClass\(\)](#).
- [style](#) ()
 - Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
 - Setter for [style\(\)](#).
- [visibility](#) ()
 - If this widget is visible.
- [setVisibility](#) (value)
 - Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
 - If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
 - Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
 - If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
 - Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
 - Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
 - Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
 - If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
 - Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
 - If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
 - Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
 - Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
 - Setter for [vstretch\(\)](#).
- [hstretch](#) ()
 - Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 - Setter for [hstretch\(\)](#).
- [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 - Setter for [busy\(\)](#).
- [registerBusy](#) ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.

- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.83.1 Detailed Description

A spacer widget for filling gaps in a layout.

It is essentially just a widget which does nothing, but respects the layouting properties like all widgets. Add them to a layout and configure them according to your sizing requirements.

1.83.2 Member Function Documentation

1.83.2.1 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.83.2.2 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.83.2.3 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.83.2.4 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.83.2.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.83.2.6 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.83.2.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.83.2.8 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.83.2.9 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.83.2.10 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.83.2.11 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.83.2.12 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.83.2.13 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.83.2.14 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.83.2.15 `effectivelyEnabled()`

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.83.2.16 `effectiveVisibility()`

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.83.2.17 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.83.2.18 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.83.2.19 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.83.2.20 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.83.2.21 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.83.2.22 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.83.2.23 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.83.2.24 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.83.2.25 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.83.2.26 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.83.2.27 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.83.2.28 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.83.2.29 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.83.2.30 `isStyleClass()`

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.83.2.31 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.83.2.32 `name()`

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.83.2.33 `nameScope()`

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.83.2.34 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.83.2.35 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.83.2.36 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.83.2.37 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.83.2.38 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.83.2.39 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.83.2.40 `OnVisibilityChanged()`

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.83.2.41 `outerSize()`

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.83.2.42 `parentWidget()`

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.83.2.43 `registerBusy()`

`registerBusy ()` [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.83.2.44 `relayout()`

`relayout ()` [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.83.2.45 `remove()`

`remove (`
 `removeconfig)` [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.83.2.46 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.83.2.47 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.83.2.48 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.49 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.50 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.51 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.52 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.53 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.54 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.55 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.83.2.56 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.57 setStrictVerticalSizing()

```
setStrictVerticalSizing (  
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.58 setStyle()

```
setStyle (  
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.59 setStyleClass()

```
setStyleClass (  
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.60 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.83.2.61 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.62 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.63 `setVstretch()`

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.83.2.64 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.83.2.65 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.83.2.66 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.83.2.67 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.83.2.68 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.83.2.69 `verticalStretchAffinity()`

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.83.2.70 `visibility()`

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.83.2.71 `vstretch()`

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

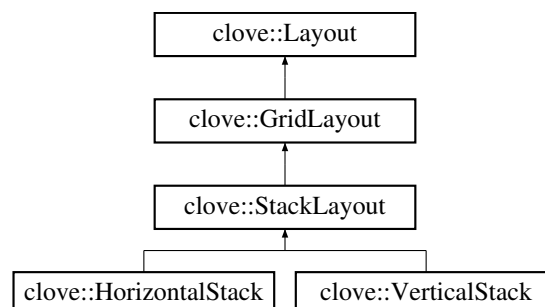
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.84 `clove::StackLayout` Class Reference

A mixin for stack layout implementations.

Inheritance diagram for `clove::StackLayout`:



Public Member Functions

- [insertRow](#) (i)
Inserts a new empty row to the grid.
- [insertColumn](#) (i)
Inserts a new empty column to the grid.
- [removeRow](#) (i)
Removes a row from the grid.
- [removeColumn](#) (i)
Removes a column from the grid.
- [children](#) ()
List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).
- [setChildren](#) (value)
Setter for [children\(\)](#).
- [addChild](#) (value)
Adds a new child widget to the layout.
- [clearChilds](#) ()
Removes all childs from the layout.

1.84.1 Detailed Description

A mixin for stack layout implementations.

1.84.2 Member Function Documentation

1.84.2.1 [addChild\(\)](#)

```
addChild (
    value ) [inherited]
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for clove::build() .
--------------	--

1.84.2.2 [children\(\)](#)

```
children ( ) [inherited]
```

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.84.2.3 clearChilds()

```
clearChilds ( ) [inherited]
```

Removes all childs from the layout.

1.84.2.4 insertColumn()

```
insertColumn (
    i ) [inherited]
```

Inserts a new empty column to the grid.

Parameters

<i>i</i>	The column index.
----------	-------------------

1.84.2.5 insertRow()

```
insertRow (
    i ) [inherited]
```

Inserts a new empty row to the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.84.2.6 removeColumn()

```
removeColumn (
    i ) [inherited]
```

Removes a column from the grid.

Parameters

<i>i</i>	The column index.
----------	-------------------

1.84.2.7 removeRow()

```
removeRow (
    i ) [inherited]
```

Removes a row from the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.84.2.8 setChildren()

```
setChildren (
    value ) [inherited]
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.85 clove::symbols Class Reference

Namespace for symbol constants.

1.85.1 Detailed Description

Namespace for symbol constants.

They make a few Unicode symbols accessible by a name.

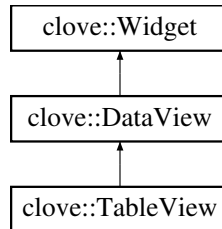
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.86 clove::TableView Class Reference

A view which shows a [clove::Datasource](#) in a table.

Inheritance diagram for `clove::TableView`:



Public Member Functions

- [datasource](#) ()
The [clove::Datasource](#) for this view.
- [setDatasource](#) (value)
Setter for [datasource\(\)](#).
- [dataViewProcessor](#) ()
An optional *function(ptr, value)* for processing a datasource value to a display string.
- [setDataViewProcessor](#) (value)
Setter for [dataViewProcessor\(\)](#).
- [cellGenerator](#) ()
A generator function for complex cell content.
- [setCellGenerator](#) (value)
Setter for [cellGenerator\(\)](#).
- [alwaysAllocateExpanderSpace](#) ()
If some space is allocated for an expander even for cells without children.
- [setAlwaysAllocateExpanderSpace](#) (value)
Setter for [alwaysAllocateExpanderSpace\(\)](#).
- [showOnlyFirstColumn](#) ()
If to show only the first column and hide the other ones.
- [setShowOnlyFirstColumn](#) (value)
Setter for [showOnlyFirstColumn\(\)](#).
- [hideExpanders](#) ()
If to hide all expanders.
- [setHideExpanders](#) (value)
Setter for [hideExpanders\(\)](#).
- [allowSelection](#) ()
If it is allowed to select nodes.
- [setAllowSelection](#) (value)
Setter for [allowSelection\(\)](#).
- [allowChecking](#) ()
If it is allowed to check cells.
- [setAllowChecking](#) (value)
Setter for [allowChecking\(\)](#).
- [granularity](#) ()
The granularity for stuff like selection and checking.

- [setGranularity](#) (value)
Setter for [granularity\(\)](#).
- [showChangeMenu](#) ()
If a menu shall be shown for making manual changes to the data source.
- [setShowChangeMenu](#) (value)
Setter for [showChangeMenu\(\)](#).
- [editOnGesture](#) ()
If the user shall be able to edit cells by double clicking/tapping them.
- [setEditOnGesture](#) (value)
Setter for [editOnGesture\(\)](#).
- [rowsResizable](#) ()
If the user shall be able to resize rows.
- [setRowsResizable](#) (value)
Setter for [rowsResizable\(\)](#).
- [columnsResizable](#) ()
If the user shall be able to resize columns.
- [setColumnsResizable](#) (value)
Setter for [columnsResizable\(\)](#).
- [selectCell](#) (ptr)
Selects a cell.
- [isCellSelected](#) (ptr)
Checks if a given cell is selected.
- [checkedCells](#) ()
Returns the checked cells as list of [clove::DatasourceValuePointer](#).
- [setCheckedCells](#) (ptrs)
Returns the checked cells.
- [isCellChecked](#) (ptr)
Checks if a given cell is checked.
- [setCellChecked](#) (ptr, val)
Sets if a given cell is checked.
- [selection](#) ()
Returns the selected item as [clove::DatasourceValuePointer](#).
- [headersource](#) ()
The [clove::Headersource](#) providing row and column header configurations.
- [setHeadersource](#) (value)
Setter for [headersource\(\)](#).
- [gridVisible](#) ()
If to show a cell grid.
- [setGridVisible](#) (value)
Setter for [gridVisible\(\)](#).
- [nodeActivationNeedsDoubleClick](#) ()
If node activation needs a double-click.
- [setNodeActivationNeedsDoubleClick](#) (value)
Setter for [nodeActivationNeedsDoubleClick\(\)](#).
- [expandCell](#) (ptr)
Expands a given cell.
- [expandCellRecursive](#) (ptr)
Expands a given cell and all parents.
- [collapseCell](#) (ptr)
Collapses a given cell.
- [isCellExpanded](#) (ptr)

- Checks if a given cell is expanded.*

 - `editCell` (ptr)

Triggers the edit mode for a cell.
- `OnSelectionChanged`

Triggered when the selection in the view changes.
- `declareProperty` (k, defaultV)

Declares a widget property.
- `getProperty` (k)

General-purpose getter for widget properties.
- `setProperty` (k, v)

General-purpose setter for widget properties.
- `bindProperty` (k, vb)

Binds a [DataBinding](#) to a property. Read [Manual](#) for details about data bindings.
- `init` (rootNameScope)

Initializes the widget.
- `doinit` ()

Executes late widget initialization (i.e. after properties are applied).
- `doinitEarly` ()

Executes early widget initialization (i.e. before properties are applied).
- `resize` ()

Applies the new widget size to internal content.
- `doresize` ()

Corrects alignments of internal elements according to the new widget size.
- `relayout` ()

Notifies the parent widget that a new geometry is required.
- `focus` ()

Sets the focus to this widget.
- `isAlive` ()

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- `computeMinimalWidth` ()

Computes the minimal width in pixel this widget needs to have.
- `computePreferredWidth` ()

Computes the preferred width in pixel this widget needs to have.
- `computeMinimalHeightForWidth` (w)

Computes the minimal height in pixel this widget needs to have for a given width.
- `computePreferredHeightForWidth` (w)

Computes the preferred height in pixel this widget needs to have for a given width.
- `getMinimalWidth` ()

Returns the minimal width in pixel this widget needs to have.
- `getPreferredWidth` ()

Returns the preferred width in pixel this widget needs to have.
- `getMinimalHeightForWidth` (w)

Returns the minimal height in pixel this widget needs to have for a given width.
- `getPreferredHeightForWidth` (w)

Returns the preferred height in pixel this widget needs to have for a given width.
- `addStyleClass` (css)

Adds the css class `css` to this widget.
- `removeStyleClass` (css)

Removes the css class `css` from this widget.
- `isStyleClass` (css)

Checks if this widget contains the css class `css`.

- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children [clove::Widget](#) instances.
- [parentWidget](#) ()
The parent [clove::Widget](#).
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)

- Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
 - If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
 - Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
 - If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
 - Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
 - Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
 - Setter for [vstretch\(\)](#).
- [hstretch](#) ()
 - Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 - Setter for [hstretch\(\)](#).
- [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 - Setter for [busy\(\)](#).
- [registerBusy](#) ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
 - Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
 - If the widget has the keyboard focus.
- [innerSize](#) ()
 - Returns inner width and height of this widget.
- [outerSize](#) ()
 - Returns outer width and height of this widget.
- [OnDestroyed](#)
 - Triggered when this widget was removed somehow.
- [OnResized](#)
 - Triggered when this widget was resized.
- [OnVisibilityChanged](#)
 - Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
 - Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
 - Triggered when a keyboard key went down.
- [OnKeyUp](#)
 - Triggered when a keyboard key came up.
- [OnKeyPress](#)
 - Triggered when a keyboard key was pressed.

1.86.1 Detailed Description

A view which shows a [clove::Datasource](#) in a table.

1.86.2 Member Function Documentation

1.86.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.86.2.2 allowChecking()

```
allowChecking ( ) [inherited]
```

If it is allowed to check cells.

This is a way to select 0..n data cells. For a single-selection, please check [allowSelection\(\)](#).

1.86.2.3 allowSelection()

```
allowSelection ( ) [inherited]
```

If it is allowed to select nodes.

This is always a single selection. For something like multi-selection, please check [allowChecking\(\)](#).

1.86.2.4 alwaysAllocateExpanderSpace()

```
alwaysAllocateExpanderSpace ( ) [inherited]
```

If some space is allocated for an expander even for cells without children.

1.86.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.86.2.6 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.86.2.7 cellGenerator()

```
cellGenerator ( ) [inherited]
```

A generator function for complex cell content.

This method is called for each cell with ([clove::Widget](#), [clove::DatasourceValuePointer](#)). It may return a widget configuration as for [clove::build\(\)](#). If it does, the cell is constructed with this widget as content. The content must define an `update(clove::Widget, clove::DatasourceValuePointer, value)` method, which takes the cell datasource value and configures the inner widget according to that.

1.86.2.8 checkedCells()

```
checkedCells ( ) [inherited]
```

Returns the checked cells as list of [clove::DatasourceValuePointer](#).

1.86.2.9 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.86.2.10 collapseCell()

```
collapseCell (
    ptr ) [inherited]
```

Collapses a given cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.86.2.11 columnsResizable()

```
columnsResizable ( ) [inherited]
```

If the user shall be able to resize columns.

1.86.2.12 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.86.2.13 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.86.2.14 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.86.2.15 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.86.2.16 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.86.2.17 datasource()

```
datasource ( ) [inherited]
```

The [clove::Datasource](#) for this view.

This provides the actual data to display. See [clove::NativeDatasource](#) for a ready-to-use implementation.

1.86.2.18 dataViewProcessor()

```
dataViewProcessor ( ) [inherited]
```

An optional `function(ptr, value)` for processing a datasource value to a display string.

1.86.2.19 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.86.2.20 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.86.2.21 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.86.2.22 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.86.2.23 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.86.2.24 editCell()

```
editCell (
    ptr ) [inherited]
```

Triggers the edit mode for a cell.

Only works if a method `_getdomcell(ir, ic, parent)` returns a dom node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.86.2.25 editOnGesture()

```
editOnGesture ( ) [inherited]
```

If the user shall be able to edit cells by double clicking/tapping them.

1.86.2.26 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.86.2.27 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.86.2.28 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.86.2.29 expandCell()

```
expandCell (
    ptr ) [inherited]
```

Expands a given cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.86.2.30 expandCellRecursive()

```
expandCellRecursive (
    ptr ) [inherited]
```

Expands a given cell and all parents.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.86.2.31 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.86.2.32 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.86.2.33 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.86.2.34 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.86.2.35 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.86.2.36 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.86.2.37 `granularity()`

```
granularity ( ) [inherited]
```

The granularity for stuff like selection and checking.

Either `'cell'` or `'row'`.

1.86.2.38 gridVisible()

`gridVisible () [inherited]`

If to show a cell grid.

1.86.2.39 hasFocus()

`hasFocus () [inherited]`

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.86.2.40 headersource()

`headersource () [inherited]`

The [clove::Headersource](#) providing row and column header configurations.

1.86.2.41 hideExpanders()

`hideExpanders () [inherited]`

If to hide all expanders.

1.86.2.42 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.86.2.43 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.86.2.44 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.86.2.45 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.86.2.46 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.86.2.47 isCellChecked()

```
isCellChecked (
    ptr ) [inherited]
```

Checks if a given cell is checked.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.86.2.48 isCellExpanded()

```
isCellExpanded (
    ptr ) [inherited]
```

Checks if a given cell is expanded.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.86.2.49 isCellSelected()

```
isCellSelected (
    ptr ) [inherited]
```

Checks if a given cell is selected.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.86.2.50 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.86.2.51 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.86.2.52 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.86.2.53 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.86.2.54 nodeActivationNeedsDoubleClick()

`nodeActivationNeedsDoubleClick () [inherited]`

If node activation needs a double-click.

Note: If you set this, you should find alternative ways for users of mobile devices!

1.86.2.55 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.86.2.56 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.86.2.57 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.86.2.58 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.86.2.59 OnPropertyChanged()

`OnPropertyChanged [inherited]`

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.86.2.60 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.86.2.61 OnSelectionChanged()

OnSelectionChanged [inherited]

Triggered when the selection in the view changes.

This is a [clove.Event](#) instance. See Manual for details.

1.86.2.62 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.86.2.63 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.86.2.64 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.86.2.65 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.86.2.66 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.86.2.67 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.86.2.68 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.86.2.69 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.86.2.70 rowsResizable()

```
rowsResizable ( ) [inherited]
```

If the user shall be able to resize rows.

1.86.2.71 selectCell()

```
selectCell (
    ptr ) [inherited]
```

Selects a cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.86.2.72 selection()

```
selection ( ) [pure virtual], [inherited]
```

Returns the selected item as [clove::DatasourceValuePointer](#).

1.86.2.73 setAllowChecking()

```
setAllowChecking (
    value ) [inherited]
```

Setter for [allowChecking\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.74 setAllowSelection()

```
setAllowSelection (
    value ) [inherited]
```

Setter for [allowSelection\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.75 setAlwaysAllocateExpanderSpace()

```
setAlwaysAllocateExpanderSpace (  
    value ) [inherited]
```

Setter for [alwaysAllocateExpanderSpace\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.76 setBusy()

```
setBusy (  
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.77 setCellChecked()

```
setCellChecked (  
    ptr,  
    val ) [inherited]
```

Sets if a given cell is checked.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>val</i>	If the cells shall be checked.

1.86.2.78 setCellGenerator()

```
setCellGenerator (
    value ) [inherited]
```

Setter for [cellGenerator\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.79 setCheckedCells()

```
setCheckedCells (
    ptrs ) [inherited]
```

Returns the checked cells.

Parameters

<i>ptrs</i>	A list of clove::DatasourceValuePointer .
-------------	---

1.86.2.80 setColumnsResizable()

```
setColumnsResizable (
    value ) [inherited]
```

Setter for [columnsResizable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.81 setDatasource()

```
setDatasource (
    value ) [inherited]
```

Setter for [datasource\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.82 setDataViewProcessor()

```
setDataViewProcessor (
    value ) [inherited]
```

Setter for [dataViewProcessor\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.83 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.84 setEditOnGesture()

```
setEditOnGesture (
    value ) [inherited]
```

Setter for [editOnGesture\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.85 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.86 setGranularity()

```
setGranularity (
    value ) [inherited]
```

Setter for [granularity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.87 setGridVisible()

```
setGridVisible (
    value ) [inherited]
```

Setter for [gridVisible\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.88 setHeadersource()

```
setHeadersource (
    value ) [inherited]
```

Setter for [headersource\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.89 setHideExpanders()

```
setHideExpanders (
    value ) [inherited]
```

Setter for [hideExpanders\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.90 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.91 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.92 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.93 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.94 setNodeActivationNeedsDoubleClick()

```
setNodeActivationNeedsDoubleClick (
    value ) [inherited]
```

Setter for [nodeActivationNeedsDoubleClick\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.95 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.86.2.96 setRowsResizable()

```
setRowsResizable (
    value ) [inherited]
```

Setter for [rowsResizable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.97 setShowChangeMenu()

```
setShowChangeMenu (
    value ) [inherited]
```

Setter for [showChangeMenu\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.98 setShowOnlyFirstColumn()

```
setShowOnlyFirstColumn (
    value ) [inherited]
```

Setter for [showOnlyFirstColumn\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.99 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.100 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.101 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.102 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.103 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.86.2.104 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.105 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.106 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.86.2.107 showChangeMenu()

```
showChangeMenu ( ) [inherited]
```

If a menu shall be shown for making manual changes to the data source.

Instead of true, it may also be a configuration object, which may contain the following:

- `item_foo_label`, `item_foo_icon`, `item_foo_isvisible`, `item_foo_action`: Specifies a menu action `foo` by four functions. Each function gets `widget`, `datasource` as parameters. Respectively they have to return a label, an icon, if it is actually visible, or have to execute the action. It is also possible to customize the existing actions `addrow`, `addrowbefore`, `addcolumn`, `addcolumnbefore`, `removerow`, `removecolumn` and edit this way.

1.86.2.108 showOnlyFirstColumn()

```
showOnlyFirstColumn ( ) [inherited]
```

If to show only the first column and hide the other ones.

1.86.2.109 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretch←Affinity() == 0`.

1.86.2.110 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.86.2.111 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.86.2.112 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.86.2.113 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.86.2.114 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.86.2.115 `visibility()`

`visibility ()` [inherited]

If this widget is visible.

One of [`clove::Visible`](#), [`clove::Invisible`](#) and [`clove::InvisibleCollapsed`](#).

See also [`effectiveVisibility\(\)`](#).

1.86.2.116 `vstretch()`

`vstretch ()` [inherited]

Alias for [`verticalStretchAffinity\(\)`](#).

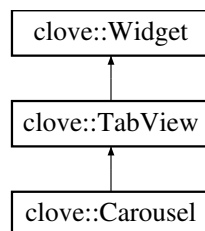
The documentation for this class was generated from the following file:

- `_meta/readme/clove.js`

1.87 `clove::TabView` Class Reference

A tabbed container.

Inheritance diagram for `clove::TabView`:



Public Member Functions

- [`tabs \(\)`](#)
The list of tabs.
- [`setTabs \(value\)`](#)
Setter for [`tabs\(\)`](#).
- [`currentTab \(\)`](#)
The index of the currently selected tab.
- [`setCurrentTab \(value\)`](#)
Setter for [`currentTab\(\)`](#).
- [`userMayAddTabs \(\)`](#)
If to show a button for adding new tabs.
- [`setUserMayAddTabs \(value\)`](#)
Setter for [`userMayAddTabs\(\)`](#).
- [`tabBarLocation \(\)`](#)

- Where to place the tab bar.*

 - [setTabBarLocation](#) (value)
Setter for [tabBarLocation\(\)](#).
 - [switchInvisibleAnimationName](#) ()
Optional animation name for switching away from a tab.
 - [setSwitchInvisibleAnimationName](#) (value)
Setter for [switchInvisibleAnimationName\(\)](#).
 - [switchVisibleAnimationName](#) ()
Optional animation name for switching to a tab.
 - [setSwitchVisibleAnimationName](#) (value)
Setter for [switchVisibleAnimationName\(\)](#).
 - [switchInvisibleAnimationDuration](#) ()
Optional animation duration (in msec) for the switch away animation.
 - [setSwitchInvisibleAnimationDuration](#) (value)
Setter for [switchInvisibleAnimationDuration\(\)](#).
 - [switchVisibleAnimationDuration](#) ()
Optional animation duration (in msec) for the switch animation.
 - [setSwitchVisibleAnimationDuration](#) (value)
Setter for [switchVisibleAnimationDuration\(\)](#).
 - [OnTabCreationRequested](#)
Triggered when the user requested a new tab.
 - [addTab](#) (config)
Adds a new tab.
 - [declareProperty](#) (k, defaultV)
Declares a widget property.
 - [getProperty](#) (k)
General-purpose getter for widget properties.
 - [setProperty](#) (k, v)
General-purpose setter for widget properties.
 - [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
 - [init](#) (rootNameScope)
Initializes the widget.
 - [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
 - [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
 - [resize](#) ()
Applies the new widget size to internal content.
 - [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
 - [relayout](#) ()
Notifies the parent widget that a new geometry is required.
 - [focus](#) ()
Sets the focus to this widget.
 - [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
 - [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
 - [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.

- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (class)
Adds the css class `class` to this widget.
- [removeStyleClass](#) (class)
Removes the css class `class` from this widget.
- [isStyleClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()

- If this widget is visible.*

 - `setVisibility` (value)
 Setter for `visibility()`.
- `doStandaloneResizing` ()
If the widget handles to resize itself as needed.

 - `setDoStandaloneResizing` (value)
 Setter for `doStandaloneResizing()`.
- `mayFocus` ()
If the widget can have the keyboard focus.

 - `setMayFocus` (value)
 Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

 - `setHorizontalStretchAffinity` (value)
 Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

 - `setVerticalStretchAffinity` (value)
 Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing` ()
If the widget is strictly forbidden to get additional horizontal space.

 - `setStrictHorizontalSizing` (value)
 Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing` ()
If the widget is strictly forbidden to get additional vertical space.

 - `setStrictVerticalSizing` (value)
 Setter for `strictVerticalSizing()`.
- `vstretch` ()
Alias for `verticalStretchAffinity()`.

 - `setVstretch` (value)
 Setter for `vstretch()`.
- `hstretch` ()
Alias for `horizontalStretchAffinity()`.

 - `setHstretch` (value)
 Setter for `hstretch()`.
- `busy` ()
If the widget is in busy state, typically resulting in a loading animation.

 - `setBusy` (value)
 Setter for `busy()`.
- `registerBusy` ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy` (token)
Drops a token and returns to normal state if no other tokens exist.
- `hasFocus` ()
If the widget has the keyboard focus.
- `innerSize` ()
Returns inner width and height of this widget.
- `outerSize` ()
Returns outer width and height of this widget.
- `OnDestroyed`
Triggered when this widget was removed somehow.

- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.87.1 Detailed Description

A tabbed container.

Typically used for grouping and folding some user interface parts together for better space efficiency and higher usability.

1.87.2 Member Function Documentation

1.87.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.87.2.2 addTab()

```
addTab (
    config )
```

Adds a new tab.

The configuration may contain the following additional keys:

- `tabLabel`: The tab header text.

- `tabIcon`: The tab icon as [clove::Icon](#).
- `tabMaybeClosedByUser`: If the user may close this tab.

This method returns a [clove::Widget](#) which can be used for getting subwidgets or removing the tab.

Parameters

<i>config</i>	A widget configuration like for clove::build() .
---------------	--

1.87.2.3 `bindProperty()`

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.87.2.4 `busy()`

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.87.2.5 `childrenWidgets()`

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.87.2.6 `computeMinimalHeightForWidth()`

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.87.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.87.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.87.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.87.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.87.2.11 `currentTab()`

```
currentTab ( )
```

The index of the currently selected tab.

1.87.2.12 `declareProperty()`

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.87.2.13 `doinit()`

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.87.2.14 `doinitEarly()`

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.87.2.15 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.87.2.16 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.87.2.17 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.87.2.18 effectiveVisibility()

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.87.2.19 enabled()

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.87.2.20 focus()

`focus () [inherited]`

Sets the focus to this widget.

1.87.2.21 getMinimalHeightForWidth()

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.87.2.22 `getMinimalWidth()`

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.87.2.23 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.87.2.24 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.87.2.25 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.87.2.26 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.87.2.27 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.87.2.28 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.87.2.29 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.87.2.30 `innerSize()`

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.87.2.31 `isAlive()`

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.87.2.32 `isStyleClass()`

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.87.2.33 `mayFocus()`

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.87.2.34 `name()`

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.87.2.35 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.87.2.36 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.87.2.37 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.87.2.38 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.87.2.39 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.87.2.40 OnPropertyChanged()

`OnPropertyChanged [inherited]`

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.87.2.41 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.87.2.42 OnTabCreationRequested()

`OnTabCreationRequested`

Triggered when the user requested a new tab.

See also [userMayAddTabs\(\)](#).

This is a [clove.Event](#) instance. See Manual for details.

1.87.2.43 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.87.2.44 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.87.2.45 parentWidget()

`parentWidget ()` [inherited]

The parent [clove::Widget](#).

1.87.2.46 registerBusy()

`registerBusy ()` [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.87.2.47 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.87.2.48 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.87.2.49 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.87.2.50 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.87.2.51 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.52 setCurrentTab()

```
setCurrentTab (
    value )
```

Setter for [currentTab\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.53 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.54 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.55 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (  
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.56 setHstretch()

```
setHstretch (  
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.57 setMayFocus()

```
setMayFocus (  
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.58 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.59 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.87.2.60 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.61 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.62 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.63 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.64 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.87.2.65 setSwitchInvisibleAnimationDuration()

```
setSwitchInvisibleAnimationDuration (
    value )
```

Setter for [switchInvisibleAnimationDuration\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.66 setSwitchInvisibleAnimationName()

```
setSwitchInvisibleAnimationName (
    value )
```

Setter for [switchInvisibleAnimationName\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.67 setSwitchVisibleAnimationDuration()

```
setSwitchVisibleAnimationDuration (
    value )
```

Setter for [switchVisibleAnimationDuration\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.68 setSwitchVisibleAnimationName()

```
setSwitchVisibleAnimationName (
    value )
```

Setter for [switchVisibleAnimationName\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.69 setTabBarLocation()

```
setTabBarLocation (  
    value )
```

Setter for [tabBarLocation\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.70 setTabs()

```
setTabs (  
    value )
```

Setter for [tabs\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.71 setUserMayAddTabs()

```
setUserMayAddTabs (  
    value )
```

Setter for [userMayAddTabs\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.72 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.73 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.74 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.87.2.75 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.87.2.76 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity()==0`.

1.87.2.77 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.87.2.78 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.87.2.79 switchInvisibleAnimationDuration()

```
switchInvisibleAnimationDuration ( )
```

Optional animation duration (in msec) for the switch away animation.

See also `clove::TabView::switchInvisibleAnimationName()`.

1.87.2.80 switchInvisibleAnimationName()

```
switchInvisibleAnimationName ( )
```

Optional animation name for switching away from a tab.

1.87.2.81 switchVisibleAnimationDuration()

```
switchVisibleAnimationDuration ( )
```

Optional animation duration (in msec) for the switch animation.

See also `clove::TabView::switchVisibleAnimationName()`.

1.87.2.82 switchVisibleAnimationName()

```
switchVisibleAnimationName ( )
```

Optional animation name for switching to a tab.

1.87.2.83 tabBarLocation()

```
tabBarLocation ( )
```

Where to place the tab bar.

Either 'top', 'left', 'bottom' or 'right'.

1.87.2.84 tabs()

```
tabs ( )
```

The list of tabs.

Each tab is a widget configuration, like for [clove::build\(\)](#), with some additional optional keys which holds infos like the tab header text. So, for example, one item in that list could be `{view:'Something', ..., tabLabel:'Tab1'}`. See also [addTab\(\)](#).

1.87.2.85 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.87.2.86 userMayAddTabs()

```
userMayAddTabs ( )
```

If to show a button for adding new tabs.

If `true`, implement a handler for `OnTabCreationRequested` and create a tab with [addTab\(\)](#) there.

1.87.2.87 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.87.2.88 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.87.2.89 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

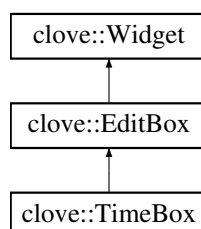
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.88 clove::TimeBox Class Reference

A user input box for time input.

Inheritance diagram for `clove::TimeBox`:



Public Member Functions

- `time ()`
The time value in the box as JavaScript Date.
- `setTime (value)`
Setter for `time()`.
- `readOnly ()`
If the edit box is read-only or editable by the user.
- `setReadOnly (value)`
Setter for `readOnly()`.
- `text ()`
The current text.
- `setText (value)`
Setter for `text()`.
- `hintText ()`
The hint text.
- `setHintText (value)`
Setter for `hintText()`.
- `autocompleteItems ()`
A [Clove.Datasource](#) with a list of autocomplete items to popup.
- `setAutocompleteItems (value)`
Setter for `autocompleteItems()`.
- `autocompleteFilter ()`
How to filter the autocomplete list.
- `setAutocompleteFilter (value)`
Setter for `autocompleteFilter()`.
- `autocompleteOpenForNoText ()`
If to show the autocomplete list when the edit box is empty.
- `setAutocompleteOpenForNoText (value)`
Setter for `autocompleteOpenForNoText()`.
- `setTextSelection (ifrom, ito)`
Selects the specified part of the text.
- `textSelection ()`
Returns the current text selection indices.
- `OnChanged`
Triggered when the text was changed.
- `OnPopupTextSelected`
Triggered when a text was selected from the popup.
- `declareProperty (k, defaultV)`
Declares a widget property.
- `getProperty (k)`
General-purpose getter for widget properties.
- `setProperty (k, v)`
General-purpose setter for widget properties.
- `bindProperty (k, vb)`
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- `init (rootNameScope)`
Initializes the widget.
- `doinit ()`
Executes late widget initialization (i.e. after properties are applied).
- `doinitEarly ()`

- Executes early widget initialization (i.e. before properties are applied).*

 - [resize](#) ()

Applies the new widget size to internal content.
 - [doresize](#) ()

Corrects alignments of internal elements according to the new widget size.
 - [relayout](#) ()

Notifies the parent widget that a new geometry is required.
 - [focus](#) ()

Sets the focus to this widget.
 - [isAlive](#) ()

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
 - [computeMinimalWidth](#) ()

Computes the minimal width in pixel this widget needs to have.
 - [computePreferredWidth](#) ()

Computes the preferred width in pixel this widget needs to have.
 - [computeMinimalHeightForWidth](#) (w)

Computes the minimal height in pixel this widget needs to have for a given width.
 - [computePreferredHeightForWidth](#) (w)

Computes the preferred height in pixel this widget needs to have for a given width.
 - [getMinimalWidth](#) ()

Returns the minimal width in pixel this widget needs to have.
 - [getPreferredWidth](#) ()

Returns the preferred width in pixel this widget needs to have.
 - [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
 - [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
 - [addStyleClass](#) (class)

Adds the css class `class` to this widget.
 - [removeStyleClass](#) (class)

Removes the css class `class` from this widget.
 - [isStyleClass](#) (class)

Checks if this widget contains the css class `class`.
 - [setStyleClassAssigned](#) (class, assigned)

Sets or unsets the css class `class` to this widget.
 - [containingNameScope](#) ()

The namespace this widget contains to.
 - [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
 - [remove](#) (removeconfig)

Removes this widget.
 - [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
 - [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
 - [childrenWidgets](#) ()

List of the children [clove::Widget](#) instances.
 - [parentWidget](#) ()

The parent [clove::Widget](#).
 - [name](#) ()

The name of the widget.

- `setName` (value)
Setter for `name()`.
- `enabled` ()
If this widget is marked as enabled (i.e. can interact with the user).
- `setEnabled` (value)
Setter for `enabled()`.
- `styleClass` ()
Custom css class(es).
- `setStyleClass` (value)
Setter for `styleClass()`.
- `style` ()
Custom css style string. You should not use that, but `styleClass()` instead.
- `setStyle` (value)
Setter for `style()`.
- `visibility` ()
If this widget is visible.
- `setVisibility` (value)
Setter for `visibility()`.
- `doStandaloneResizing` ()
If the widget handles to resize itself as needed.
- `setDoStandaloneResizing` (value)
Setter for `doStandaloneResizing()`.
- `mayFocus` ()
If the widget can have the keyboard focus.
- `setMayFocus` (value)
Setter for `mayFocus()`.
- `horizontalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity` (value)
Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity` ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity` (value)
Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing` ()
If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing` (value)
Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing` ()
If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing` (value)
Setter for `strictVerticalSizing()`.
- `vstretch` ()
Alias for `verticalStretchAffinity()`.
- `setVstretch` (value)
Setter for `vstretch()`.
- `hstretch` ()
Alias for `horizontalStretchAffinity()`.
- `setHstretch` (value)
Setter for `hstretch()`.
- `busy` ()

- If the widget is in busy state, typically resulting in a loading animation.*
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.88.1 Detailed Description

A user input box for time input.

The actual behavior depends on the browser.

1.88.2 Member Function Documentation

1.88.2.1 [addStyleClass\(\)](#)

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.88.2.2 autoCompleteFilter()

`autoCompleteFilter () [inherited]`

How to filter the autocomplete list.

Either 'startswith', 'contains', `function(itemtext, boxtext)` or undefined.

1.88.2.3 autoCompleteItems()

`autoCompleteItems () [inherited]`

A [clove.DataSource](#) with a list of autocomplete items to popup.

It is allowed to observe the `text` in order to generate live content for this list.

1.88.2.4 autoCompleteOpenForNoText()

`autoCompleteOpenForNoText () [inherited]`

If to show the autocomplete list when the edit box is empty.

1.88.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.88.2.6 busy()

`busy () [inherited]`

If the widget is in busy state, typically resulting in a loading animation.

1.88.2.7 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.88.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.88.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.88.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.88.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.88.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.88.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.88.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.88.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.88.2.16 doresize()

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.88.2.17 doStandaloneResizing()

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.88.2.18 effectivelyEnabled()

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.88.2.19 effectiveVisibility()

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.88.2.20 enabled()

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.88.2.21 focus()

`focus () [inherited]`

Sets the focus to this widget.

1.88.2.22 getMinimalHeightForWidth()

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.88.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.88.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.88.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.88.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.88.2.27 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.88.2.28 hintText()

```
hintText ( ) [inherited]
```

The hint text.

Typically contains something like a label text. It is shown as a hint when the box is empty.

1.88.2.29 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.88.2.30 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.88.2.31 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.88.2.32 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.88.2.33 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.88.2.34 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class *class*.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.88.2.35 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.88.2.36 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.88.2.37 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.88.2.38 OnChanged()

`OnChanged [inherited]`

Triggered when the text was changed.

This is a [clove.Event](#) instance. See Manual for details.

1.88.2.39 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.88.2.40 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.88.2.41 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.88.2.42 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.88.2.43 OnPopupTextSelected()

OnPopupTextSelected [inherited]

Triggered when a text was selected from the popup.

This is a [clove.Event](#) instance. See Manual for details.

1.88.2.44 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.88.2.45 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.88.2.46 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.88.2.47 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.88.2.48 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.88.2.49 readOnly()

```
readOnly ( ) [inherited]
```

If the edit box is read-only or editable by the user.

1.88.2.50 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.88.2.51 relayayout()

```
relayayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.88.2.52 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.88.2.53 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.88.2.54 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.88.2.55 `setAutocompletionFilter()`

```
setAutocompletionFilter (
    value ) [inherited]
```

Setter for [autocompletionFilter\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.56 `setAutocompletionItems()`

```
setAutocompletionItems (
    value ) [inherited]
```

Setter for [autocompletionItems\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.57 `setAutocompletionOpenForNoText()`

```
setAutocompletionOpenForNoText (
    value ) [inherited]
```

Setter for [autocompletionOpenForNoText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.58 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.59 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.60 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.61 `setHintText()`

```
setHintText (
    value ) [inherited]
```

Setter for [hintText\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.62 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.63 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.64 `setMayFocus()`

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.65 setName()

```
setName (  
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.66 setProperty()

```
setProperty (  
    k,  
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.88.2.67 setReadOnly()

```
setReadOnly (  
    value ) [inherited]
```

Setter for [readOnly\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.68 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (  
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.69 setStrictVerticalSizing()

```
setStrictVerticalSizing (  
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.70 setStyle()

```
setStyle (  
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.71 setStyleClass()

```
setStyleClass (  
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.72 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.88.2.73 `setText()`

```
setText (
    value ) [inherited]
```

Setter for [text\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.74 `setTextSelection()`

```
setTextSelection (
    ifrom,
    ito ) [inherited]
```

Selects the specified part of the text.

Parameters

<i>ifrom</i>	The selection begin index.
<i>ito</i>	The selection end index.

1.88.2.75 setTime()

```
setTime (
    value )
```

Setter for [time\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.76 setVerticalStretchAffinity()

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.77 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.78 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.88.2.79 strictHorizontalSizing()

`strictHorizontalSizing () [inherited]`

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.88.2.80 strictVerticalSizing()

`strictVerticalSizing () [inherited]`

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.88.2.81 style()

`style () [inherited]`

Custom css style string. You should not use that, but `styleClass()` instead.

1.88.2.82 styleClass()

`styleClass () [inherited]`

Custom css class(es).

1.88.2.83 text()

`text () [inherited]`

The current text.

1.88.2.84 textSelection()

```
textSelection ( ) [inherited]
```

Returns the current text selection indices.

1.88.2.85 time()

```
time ( )
```

The time value in the box as JavaScript Date.

1.88.2.86 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.88.2.87 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.88.2.88 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.88.2.89 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

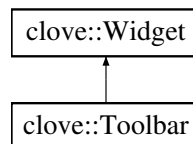
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.89 clove::Toolbar Class Reference

A toolbar has header texts and a menu bar in a perceptible visual bar, typically used on top of a user interface with all available width.

Inheritance diagram for clove::Toolbar:



Public Member Functions

- [actions](#) ()
Menu actions.
- [setActions](#) (value)
Setter for [actions\(\)](#).
- [head1](#) ()
The 1st header text.
- [setHead1](#) (value)
Setter for [head1\(\)](#).
- [head2](#) ()
The 2nd header text.
- [setHead2](#) (value)
Setter for [head2\(\)](#).
- [headControl](#) ()
An optional widget for arbitrary control purposes as widget configuration like for [clove::build\(\)](#). It's displayed below the menu bar in full width.
- [setHeadControl](#) (value)
Setter for [headControl\(\)](#).
- [headControlWidget](#) ()
Returns the control widget as [clove::Widget](#) (or undefined if no [headControl\(\)](#) is set).
- [icon](#) ()
The header icon as [clove::Icon](#).
- [setIcon](#) (value)
Setter for [icon\(\)](#).
- [OnActionTriggered](#)

- Triggered when a menu action was chosen by the user for execution.*
- `declareProperty` (k, defaultV)

Declares a widget property.
- `getProperty` (k)

General-purpose getter for widget properties.
- `setProperty` (k, v)

General-purpose setter for widget properties.
- `bindProperty` (k, vb)

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- `init` (rootNameScope)

Initializes the widget.
- `doinit` ()

Executes late widget initialization (i.e. after properties are applied).
- `doinitEarly` ()

Executes early widget initialization (i.e. before properties are applied).
- `resize` ()

Applies the new widget size to internal content.
- `doresize` ()

Corrects alignments of internal elements according to the new widget size.
- `relayout` ()

Notifies the parent widget that a new geometry is required.
- `focus` ()

Sets the focus to this widget.
- `isAlive` ()

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- `computeMinimalWidth` ()

Computes the minimal width in pixel this widget needs to have.
- `computePreferredWidth` ()

Computes the preferred width in pixel this widget needs to have.
- `computeMinimalHeightForWidth` (w)

Computes the minimal height in pixel this widget needs to have for a given width.
- `computePreferredHeightForWidth` (w)

Computes the preferred height in pixel this widget needs to have for a given width.
- `getMinimalWidth` ()

Returns the minimal width in pixel this widget needs to have.
- `getPreferredWidth` ()

Returns the preferred width in pixel this widget needs to have.
- `getMinimalHeightForWidth` (w)

Returns the minimal height in pixel this widget needs to have for a given width.
- `getPreferredHeightForWidth` (w)

Returns the preferred height in pixel this widget needs to have for a given width.
- `addStyleClass` (css)

Adds the css class `css` to this widget.
- `removeStyleClass` (css)

Removes the css class `css` from this widget.
- `isStyleClass` (css)

Checks if this widget contains the css class `css`.
- `setStyleClassAssigned` (css, assigned)

Sets or unsets the css class `css` to this widget.
- `containingNameScope` ()

The namespace this widget contains to.

- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children [clove::Widget](#) instances.
- [parentWidget](#) ()
The parent [clove::Widget](#).
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)

- Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
 - If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
 - Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
 - Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
 - Setter for [vstretch\(\)](#).
- [hstretch](#) ()
 - Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 - Setter for [hstretch\(\)](#).
- [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 - Setter for [busy\(\)](#).
- [registerBusy](#) ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
 - Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
 - If the widget has the keyboard focus.
- [innerSize](#) ()
 - Returns inner width and height of this widget.
- [outerSize](#) ()
 - Returns outer width and height of this widget.
- [OnDestroyed](#)
 - Triggered when this widget was removed somehow.
- [OnResized](#)
 - Triggered when this widget was resized.
- [OnVisibilityChanged](#)
 - Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
 - Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
 - Triggered when a keyboard key went down.
- [OnKeyUp](#)
 - Triggered when a keyboard key came up.
- [OnKeyPress](#)
 - Triggered when a keyboard key was pressed.

1.89.1 Detailed Description

A toolbar has header texts and a menu bar in a perceptible visual bar, typically used on top of a user interface with all available width.

1.89.2 Member Function Documentation

1.89.2.1 actions()

```
actions ( )
```

Menu actions.

See [clove::Menubar::actions\(\)](#).

1.89.2.2 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.89.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.89.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.89.2.5 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.89.2.6 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.89.2.7 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.89.2.8 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.89.2.9 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.89.2.10 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.89.2.11 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.89.2.12 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.89.2.13 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.89.2.14 `doresize()`

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.89.2.15 `doStandaloneResizing()`

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.89.2.16 `effectivelyEnabled()`

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.89.2.17 `effectiveVisibility()`

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.89.2.18 `enabled()`

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.89.2.19 `focus()`

`focus () [inherited]`

Sets the focus to this widget.

1.89.2.20 `getMinimalHeightForWidth()`

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.89.2.21 `getMinimalWidth()`

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.89.2.22 `getPreferredHeightForWidth()`

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.89.2.23 `getPreferredWidth()`

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.89.2.24 `getProperty()`

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.89.2.25 hasFocus()

`hasFocus () [inherited]`

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.89.2.26 head1()

`head1 ()`

The 1st header text.

1.89.2.27 head2()

`head2 ()`

The 2nd header text.

1.89.2.28 headControl()

`headControl ()`

An optional widget for arbitrary control purposes as widget configuration like for [clove::build\(\)](#). It's displayed below the menu bar in full width.

1.89.2.29 headControlWidget()

`headControlWidget ()`

Returns the control widget as [clove::Widget](#) (or undefined if no [headControl\(\)](#) is set).

1.89.2.30 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.89.2.31 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.89.2.32 icon()

`icon ()`

The header icon as [clove::Icon](#).

1.89.2.33 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.89.2.34 innerSize()

`innerSize () [inherited]`

Returns inner width and height of this widget.

1.89.2.35 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.89.2.36 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.89.2.37 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.89.2.38 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.89.2.39 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.89.2.40 OnActionTriggered()

`OnActionTriggered`

Triggered when a menu action was chosen by the user for execution.

The event arguments contain the selected action name in `action`.

This is a [clove.Event](#) instance. See Manual for details.

1.89.2.41 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.89.2.42 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.89.2.43 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.89.2.44 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.89.2.45 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.89.2.46 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [Clove.Event](#) instance. See Manual for details.

1.89.2.47 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [Clove.Event](#) instance. See Manual for details.

1.89.2.48 outerSize()

`outerSize ()` [inherited]

Returns outer width and height of this widget.

1.89.2.49 parentWidget()

`parentWidget ()` [inherited]

The parent [Clove::Widget](#).

1.89.2.50 registerBusy()

`registerBusy ()` [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.89.2.51 relayout()

`relayout ()` [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [Clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.89.2.52 remove()

`remove (
 removeconfig)` [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [Clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.89.2.53 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class *class* from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.89.2.54 resize()

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.89.2.55 setActions()

```
setActions (
    value )
```

Setter for [actions\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.56 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.57 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.58 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.59 setHead1()

```
setHead1 (
    value )
```

Setter for [head1\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.60 setHead2()

```
setHead2 (
    value )
```

Setter for [head2\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.61 setHeadControl()

```
setHeadControl (
    value )
```

Setter for [headControl\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.62 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.63 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.64 setIcon()

```
setIcon (  
    value )
```

Setter for [icon\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.65 setMayFocus()

```
setMayFocus (  
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.66 setName()

```
setName (  
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.67 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.89.2.68 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.69 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.70 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.71 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.72 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.89.2.73 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.74 setVisibility()

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.75 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.89.2.76 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.89.2.77 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.89.2.78 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.89.2.79 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.89.2.80 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.89.2.81 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.89.2.82 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.89.2.83 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

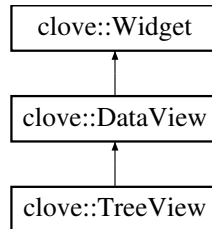
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.90 clove::TreeView Class Reference

A view which shows a [clove::Datasource](#) in a tree.

Inheritance diagram for clove::TreeView:



Public Member Functions

- [datasource](#) ()
The [clove::Datasource](#) for this view.
- [setDatasource](#) (value)
Setter for [datasource\(\)](#).
- [dataViewProcessor](#) ()
An optional *function(ptr, value)* for processing a datasource value to a display string.
- [setDataViewProcessor](#) (value)
Setter for [dataViewProcessor\(\)](#).
- [cellGenerator](#) ()
A generator function for complex cell content.
- [setCellGenerator](#) (value)
Setter for [cellGenerator\(\)](#).
- [alwaysAllocateExpanderSpace](#) ()
If some space is allocated for an expander even for cells without children.
- [setAlwaysAllocateExpanderSpace](#) (value)
Setter for [alwaysAllocateExpanderSpace\(\)](#).
- [showOnlyFirstColumn](#) ()
If to show only the first column and hide the other ones.
- [setShowOnlyFirstColumn](#) (value)
Setter for [showOnlyFirstColumn\(\)](#).
- [hideExpanders](#) ()
If to hide all expanders.
- [setHideExpanders](#) (value)
Setter for [hideExpanders\(\)](#).
- [allowSelection](#) ()
If it is allowed to select nodes.
- [setAllowSelection](#) (value)
Setter for [allowSelection\(\)](#).
- [allowChecking](#) ()
If it is allowed to check cells.
- [setAllowChecking](#) (value)
Setter for [allowChecking\(\)](#).
- [granularity](#) ()
The granularity for stuff like selection and checking.

- [setGranularity](#) (value)
Setter for [granularity\(\)](#).
- [showChangeMenu](#) ()
If a menu shall be shown for making manual changes to the data source.
- [setShowChangeMenu](#) (value)
Setter for [showChangeMenu\(\)](#).
- [editOnGesture](#) ()
If the user shall be able to edit cells by double clicking/tapping them.
- [setEditOnGesture](#) (value)
Setter for [editOnGesture\(\)](#).
- [rowsResizable](#) ()
If the user shall be able to resize rows.
- [setRowsResizable](#) (value)
Setter for [rowsResizable\(\)](#).
- [columnsResizable](#) ()
If the user shall be able to resize columns.
- [setColumnsResizable](#) (value)
Setter for [columnsResizable\(\)](#).
- [selectCell](#) (ptr)
Selects a cell.
- [isCellSelected](#) (ptr)
Checks if a given cell is selected.
- [checkedCells](#) ()
Returns the checked cells as list of [clove::DatasourceValuePointer](#).
- [setCheckedCells](#) (ptrs)
Returns the checked cells.
- [isCellChecked](#) (ptr)
Checks if a given cell is checked.
- [setCellChecked](#) (ptr, val)
Sets if a given cell is checked.
- [selection](#) ()
Returns the selected item as [clove::DatasourceValuePointer](#).
- [headersource](#) ()
The [clove::Headersource](#) providing row and column header configurations.
- [setHeadersource](#) (value)
Setter for [headersource\(\)](#).
- [gridVisible](#) ()
If to show a cell grid.
- [setGridVisible](#) (value)
Setter for [gridVisible\(\)](#).
- [nodeActivationNeedsDoubleClick](#) ()
If node activation needs a double-click.
- [setNodeActivationNeedsDoubleClick](#) (value)
Setter for [nodeActivationNeedsDoubleClick\(\)](#).
- [expandCell](#) (ptr)
Expands a given cell.
- [expandCellRecursive](#) (ptr)
Expands a given cell and all parents.
- [collapseCell](#) (ptr)
Collapses a given cell.
- [isCellExpanded](#) (ptr)

- Checks if a given cell is expanded.*

 - [editCell](#) (ptr)

Triggers the edit mode for a cell.
- [OnSelectionChanged](#)

Triggered when the selection in the view changes.
- [declareProperty](#) (k, defaultV)

Declares a widget property.
- [getProperty](#) (k)

General-purpose getter for widget properties.
- [setProperty](#) (k, v)

General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)

Initializes the widget.
- [doinit](#) ()

Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()

Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()

Applies the new widget size to internal content.
- [doresize](#) ()

Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()

Notifies the parent widget that a new geometry is required.
- [focus](#) ()

Sets the focus to this widget.
- [isAlive](#) ()

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()

Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()

Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)

Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)

Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()

Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()

Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)

Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)

Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)

Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)

Removes the css class `css` from this widget.
- [isStyleClass](#) (css)

Checks if this widget contains the css class `css`.

- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
Setter for [style\(\)](#).
- [visibility](#) ()
If this widget is visible.
- [setVisibility](#) (value)
Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)

- Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
 - If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
 - Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
 - If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
 - Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
 - Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
 - Setter for [vstretch\(\)](#).
- [hstretch](#) ()
 - Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 - Setter for [hstretch\(\)](#).
- [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
 - Setter for [busy\(\)](#).
- [registerBusy](#) ()
 - Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
 - Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
 - If the widget has the keyboard focus.
- [innerSize](#) ()
 - Returns inner width and height of this widget.
- [outerSize](#) ()
 - Returns outer width and height of this widget.
- [OnDestroyed](#)
 - Triggered when this widget was removed somehow.
- [OnResized](#)
 - Triggered when this widget was resized.
- [OnVisibilityChanged](#)
 - Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
 - Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
 - Triggered when a keyboard key went down.
- [OnKeyUp](#)
 - Triggered when a keyboard key came up.
- [OnKeyPress](#)
 - Triggered when a keyboard key was pressed.

1.90.1 Detailed Description

A view which shows a [clove::Datasource](#) in a tree.

1.90.2 Member Function Documentation

1.90.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.90.2.2 allowChecking()

```
allowChecking ( ) [inherited]
```

If it is allowed to check cells.

This is a way to select 0..n data cells. For a single-selection, please check [allowSelection\(\)](#).

1.90.2.3 allowSelection()

```
allowSelection ( ) [inherited]
```

If it is allowed to select nodes.

This is always a single selection. For something like multi-selection, please check [allowChecking\(\)](#).

1.90.2.4 alwaysAllocateExpanderSpace()

```
alwaysAllocateExpanderSpace ( ) [inherited]
```

If some space is allocated for an expander even for cells without children.

1.90.2.5 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.90.2.6 busy()

`busy () [inherited]`

If the widget is in busy state, typically resulting in a loading animation.

1.90.2.7 cellGenerator()

`cellGenerator () [inherited]`

A generator function for complex cell content.

This method is called for each cell with ([clove::Widget](#), [clove::DatasourceValuePointer](#)). It may return a widget configuration as for [clove::build\(\)](#). If it does, the cell is constructed with this widget as content. The content must define an `update(clove::Widget, clove::DatasourceValuePointer, value)` method, which takes the cell datasource value and configures the inner widget according to that.

1.90.2.8 checkedCells()

`checkedCells () [inherited]`

Returns the checked cells as list of [clove::DatasourceValuePointer](#).

1.90.2.9 childrenWidgets()

`childrenWidgets () [inherited]`

List of the children [clove::Widget](#) instances.

1.90.2.10 collapseCell()

`collapseCell (
 ptr) [inherited]`

Collapses a given cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.90.2.11 columnsResizable()

```
columnsResizable ( ) [inherited]
```

If the user shall be able to resize columns.

1.90.2.12 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.90.2.13 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.90.2.14 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.90.2.15 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.90.2.16 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.90.2.17 datasource()

```
datasource ( ) [inherited]
```

The [clove::Datasource](#) for this view.

This provides the actual data to display. See [clove::NativeDatasource](#) for a ready-to-use implementation.

1.90.2.18 dataViewProcessor()

```
dataViewProcessor ( ) [inherited]
```

An optional `function(ptr, value)` for processing a datasource value to a display string.

1.90.2.19 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.90.2.20 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.90.2.21 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.90.2.22 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.90.2.23 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.90.2.24 editCell()

```
editCell (
    ptr ) [inherited]
```

Triggers the edit mode for a cell.

Only works if a method `_getdomcell(ir, ic, parent)` returns a dom node.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.90.2.25 editOnGesture()

```
editOnGesture ( ) [inherited]
```

If the user shall be able to edit cells by double clicking/tapping them.

1.90.2.26 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.90.2.27 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.90.2.28 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.90.2.29 expandCell()

```
expandCell (
    ptr ) [inherited]
```

Expands a given cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.90.2.30 `expandCellRecursive()`

```
expandCellRecursive (
    ptr ) [inherited]
```

Expands a given cell and all parents.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.90.2.31 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.90.2.32 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.90.2.33 `getMinimalWidth()`

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.90.2.34 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.90.2.35 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.90.2.36 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.90.2.37 granularity()

```
granularity ( ) [inherited]
```

The granularity for stuff like selection and checking.

Either `'cell'` or `'row'`.

1.90.2.38 gridVisible()

`gridVisible () [inherited]`

If to show a cell grid.

1.90.2.39 hasFocus()

`hasFocus () [inherited]`

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.90.2.40 headersource()

`headersource () [inherited]`

The [Clove::Headersource](#) providing row and column header configurations.

1.90.2.41 hideExpanders()

`hideExpanders () [inherited]`

If to hide all expanders.

1.90.2.42 horizontalStretchAffinity()

`horizontalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.90.2.43 hstretch()

`hstretch () [inherited]`

Alias for [horizontalStretchAffinity\(\)](#).

1.90.2.44 init()

`init (
 rootNameScope) [inherited]`

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.90.2.45 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.90.2.46 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.90.2.47 isCellChecked()

```
isCellChecked (
    ptr ) [inherited]
```

Checks if a given cell is checked.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.90.2.48 isCellExpanded()

```
isCellExpanded (
    ptr ) [inherited]
```

Checks if a given cell is expanded.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.90.2.49 isCellSelected()

```
isCellSelected (
    ptr ) [inherited]
```

Checks if a given cell is selected.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.90.2.50 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.90.2.51 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.90.2.52 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.90.2.53 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.90.2.54 nodeActivationNeedsDoubleClick()

`nodeActivationNeedsDoubleClick () [inherited]`

If node activation needs a double-click.

Note: If you set this, you should find alternative ways for users of mobile devices!

1.90.2.55 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.90.2.56 OnKeyDown()

`OnKeyDown [inherited]`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.90.2.57 OnKeyPress()

`OnKeyPress [inherited]`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.90.2.58 OnKeyUp()

`OnKeyUp [inherited]`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.90.2.59 OnPropertyChanged()

`OnPropertyChanged [inherited]`

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.90.2.60 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.90.2.61 OnSelectionChanged()

OnSelectionChanged [inherited]

Triggered when the selection in the view changes.

This is a [clove.Event](#) instance. See Manual for details.

1.90.2.62 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.90.2.63 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.90.2.64 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.90.2.65 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.90.2.66 `relayout()`

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.90.2.67 `remove()`

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.90.2.68 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.90.2.69 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.90.2.70 rowsResizable()

```
rowsResizable ( ) [inherited]
```

If the user shall be able to resize rows.

1.90.2.71 selectCell()

```
selectCell (
    ptr ) [inherited]
```

Selects a cell.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
------------	---

1.90.2.72 selection()

```
selection ( ) [pure virtual], [inherited]
```

Returns the selected item as [clove::DatasourceValuePointer](#).

1.90.2.73 setAllowChecking()

```
setAllowChecking (
    value ) [inherited]
```

Setter for [allowChecking\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.74 setAllowSelection()

```
setAllowSelection (
    value ) [inherited]
```

Setter for [allowSelection\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.75 setAlwaysAllocateExpanderSpace()

```
setAlwaysAllocateExpanderSpace (
    value ) [inherited]
```

Setter for [alwaysAllocateExpanderSpace\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.76 setBusy()

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.77 setCellChecked()

```
setCellChecked (
    ptr,
    val ) [inherited]
```

Sets if a given cell is checked.

Parameters

<i>ptr</i>	The clove::DatasourceValuePointer .
<i>val</i>	If the cells shall be checked.

1.90.2.78 setCellGenerator()

```
setCellGenerator (
    value ) [inherited]
```

Setter for [cellGenerator\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.79 setCheckedCells()

```
setCheckedCells (
    ptrs ) [inherited]
```

Returns the checked cells.

Parameters

<i>ptrs</i>	A list of clove::DatasourceValuePointer .
-------------	---

1.90.2.80 setColumnsResizable()

```
setColumnsResizable (
    value ) [inherited]
```

Setter for [columnsResizable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.81 setDatasource()

```
setDatasource (
    value ) [inherited]
```

Setter for [datasource\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.82 setDataViewProcessor()

```
setDataViewProcessor (
    value ) [inherited]
```

Setter for [dataViewProcessor\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.83 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.84 setEditOnGesture()

```
setEditOnGesture (
    value ) [inherited]
```

Setter for [editOnGesture\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.85 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.86 `setGranularity()`

```
setGranularity (
    value ) [inherited]
```

Setter for [granularity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.87 `setGridVisible()`

```
setGridVisible (
    value ) [inherited]
```

Setter for [gridVisible\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.88 `setHeadersource()`

```
setHeadersource (
    value ) [inherited]
```

Setter for [headersource\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.89 setHideExpanders()

```
setHideExpanders (  
    value ) [inherited]
```

Setter for [hideExpanders\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.90 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (  
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.91 setHstretch()

```
setHstretch (  
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.92 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.93 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.94 setNodeActivationNeedsDoubleClick()

```
setNodeActivationNeedsDoubleClick (
    value ) [inherited]
```

Setter for [nodeActivationNeedsDoubleClick\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.95 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.90.2.96 setRowsResizable()

```
setRowsResizable (
    value ) [inherited]
```

Setter for [rowsResizable\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.97 setShowChangeMenu()

```
setShowChangeMenu (
    value ) [inherited]
```

Setter for [showChangeMenu\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.98 setShowOnlyFirstColumn()

```
setShowOnlyFirstColumn (
    value ) [inherited]
```

Setter for [showOnlyFirstColumn\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.99 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.100 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.101 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.102 setStyleClass()

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.103 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.90.2.104 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.105 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.106 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.90.2.107 showChangeMenu()

```
showChangeMenu ( ) [inherited]
```

If a menu shall be shown for making manual changes to the data source.

Instead of true, it may also be a configuration object, which may contain the following:

- `item_foo_label`, `item_foo_icon`, `item_foo_isvisible`, `item_foo_action`: Specifies a menu action `foo` by four functions. Each function gets `widget`, `datasource` as parameters. Respectively they have to return a label, an icon, if it is actually visible, or have to execute the action. It is also possible to customize the existing actions `addrow`, `addrowbefore`, `addcolumn`, `addcolumnbefore`, `removerow`, `removecolumn` and edit this way.

1.90.2.108 showOnlyFirstColumn()

```
showOnlyFirstColumn ( ) [inherited]
```

If to show only the first column and hide the other ones.

1.90.2.109 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretch←Affinity() == 0`.

1.90.2.110 strictVerticalSizing()

`strictVerticalSizing () [inherited]`

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.90.2.111 style()

`style () [inherited]`

Custom css style string. You should not use that, but `styleClass()` instead.

1.90.2.112 styleClass()

`styleClass () [inherited]`

Custom css class(es).

1.90.2.113 unregisterBusy()

`unregisterBusy (
 token) [inherited]`

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by <code>registerBusy()</code> .
--------------	--

1.90.2.114 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.90.2.115 visibility()

visibility () [inherited]

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.90.2.116 vstretch()

vstretch () [inherited]

Alias for [verticalStretchAffinity\(\)](#).

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.91 clove::utils Class Reference

Namespace for some utilities.

Public Member Functions

- [main_parts_resized](#) (reason)
Call this method if resizing is required due to external situations.
- [applyDefaultsToConfig](#) (config, defaults)
Returns a copy of `config` with the additional members from `defaults`, wherever there was no value in `config`.
- [addOnDestroyHandler](#) (domnode, handler)
Listen for the removal of a node in the dom tree and eventually execute a handler function.
- [addOnDragHandler](#) (domnode, dragbeginhandler, draghandler, dragendhandler)
Listen for a drag event of a node and eventually execute handler functions.
- [addOnDoubleClickTapHandler](#) (domnode, handler)
Listen for double click/tap event of a node and eventually execute a handler function.
- [removeOnDoubleClickTapHandler](#) (domnode, handler)
Remove a double click/tap event handler.
- [arraysum](#) (x)
Computes the sum of all elements in an Array.
- [findWidgetForDomNode](#) (node)
Finds the (most inner) [clove::Widget](#) a given dom node contains to.
- [getExtraSize](#) (node, withMargin)
Computes the extra size of a dom node.
- [getInnerSize](#) (node)
Computes the inner size of a dom node.
- [getOuterSize](#) (node, withMargin)
Computes the outer size of a dom node.

- [suspendResizing](#) ()
Temporarily suspends all user interface resizing.
- [resumeResizing](#) (token)
Resumes the user interface resizing after it was suspended with [clove::utils::suspendResizing](#).
- [cssLengthToPixels](#) (v, axis)
Translates a css length specification string to a numeric pixel value.
- [insertToArray](#) (array, i, v)
Inserts a value into an array.
- [popup](#) (config, showconfig)
Shows a popup.
- [isDescendantOf](#) (d, p)
Checks if d is a descendant of p in the html dom node.
- [arrayToDatasource](#) (array)
Builds a [clove::Datasource](#) containing the same content as the input array.
- [connectDatasourceToWidget](#) (widget, datasource, dspropname, rootptrpropname, inserthandlername, removehandlername, updatehandlername, afterdisconnectedfct, beforeconnectedfct, afterconnectedfct)
Connects a [clove::Datasource](#) to an internal widget interface (also removes the old connection).
- [animateNode](#) (node, animationname, animationduration, animationendedfct)
Plays a css animation on a html dom node for one iteration and fires a callback afterwards.
- [setStyleClassAssigned](#) (node, class, assigned)
Sets or unsets a css class to an html dom node.
- [deferLoading](#) (eload)
Defers loading of Clove by an event.
- [runOnLoaded](#) (fct)
Runs a function as soon as possible, but not before Clove has finished loading.
- [getFullPathForLoadedResource](#) (tgtname, elementtype, elementsrccattrname)
Returns the full path for an already loaded script or stylesheet with known file name.
- [tryLoadScript](#) (scriptpath, config)
Tries to load another javascript.
- [tryLoadStyle](#) (stylepath, config)
Tries to load a css stylesheet.
- [OnMainResized](#)
Triggered when the main view resized.

1.91.1 Detailed Description

Namespace for some utilities.

1.91.2 Member Function Documentation

1.91.2.1 addOnDestroyHandler()

```
addOnDestroyHandler (
    domnode,
    handler )
```

Listen for the removal of a node in the dom tree and eventually execute a handler function.

Parameters

<i>domnode</i>	The node in the dom tree to observe.
<i>handler</i>	The function to execute on removal of <i>domnode</i> .

1.91.2.2 addOnDoubleClickTapHandler()

```
addOnDoubleClickTapHandler (
    domnode,
    handler )
```

Listen for double click/tap event of a node and eventually execute a handler function.

This is like a pure double click event, but also supports touch.

See also [clove::utils::removeOnDoubleClickTapHandler\(\)](#).

Parameters

<i>domnode</i>	The node in the dom tree to observe.
<i>handler</i>	The function to execute on double click/tap.

1.91.2.3 addOnDragHandler()

```
addOnDragHandler (
    domnode,
    dragbeginhandler,
    draghandler,
    dragendhandler )
```

Listen for a drag event of a node and eventually execute handler functions.

A drag consists of mouse/finger down, moving and releasing.

Parameters

<i>domnode</i>	The node in the dom tree to observe.
<i>dragbeginhandler</i>	The function to execute when a drag operation begins.
<i>draghandler</i>	The function to execute while moving (with relative coordinates as parameters).
<i>dragendhandler</i>	The function to execute when a drag operation ends.

1.91.2.4 animateNode()

```
animateNode (
    node,
    animationname,
    animationduration,
    animationendedfct )
```

Plays a css animation on a html dom node for one iteration and fires a callback afterwards.

Parameters

<i>node</i>	The html dom node to animate.
<i>animationname</i>	The css animation name.
<i>animationduration</i>	The animation duration in milliseconds.
<i>animationendedfct</i>	a callback <code>function()</code> called when the animation ends.

1.91.2.5 applyDefaultsToConfig()

```
applyDefaultsToConfig (
    config,
    defaults )
```

Returns a copy of `config` with the additional members from `defaults`, wherever there was no value in `config`.

Parameters

<i>config</i>	The source object.
<i>defaults</i>	Default values for members to apply to the result wherever they are missing.

1.91.2.6 arraysum()

```
arraysum (
    x )
```

Computes the sum of all elements in an Array.

Parameters

<i>x</i>	The Array to sum up.
----------	----------------------

1.91.2.7 arrayToDatasource()

```
arrayToDatasource (
    array )
```

Builds a [clove::Datasource](#) containing the same content as the input array.

Parameters

<i>array</i>	The input array.
--------------	------------------

1.91.2.8 connectDatasourceToWidget()

```
connectDatasourceToWidget (
    widget,
    datasource,
    dspropname,
    rootptrpropname,
    inserthandlername,
    removehandlername,
    updatehandlername,
    afterdisconnectedfct,
    beforeconnectedfct,
    afterconnectedfct )
```

Connects a [clove::Datasource](#) to an internal widget interface (also removes the old connection).

Connects a datasource to a widget.

This method is solely used by the inner implementation of a widget for using a datasource. It is not useful to call it in any other situation!

Parameters

<i>widget</i>	A clove::Widget .
<i>datasource</i>	A clove::Datasource .
<i>dspropname</i>	The property name for storing the current datasource in the widget.
<i>rootptrpropname</i>	The property name to get the root clove::DatasourceValuePointer from the widget.
<i>inserthandlername</i>	The name of the widget's insert handler method.
<i>removehandlername</i>	The name of the widget's remove handler method.
<i>updatehandlername</i>	The name of the widget's update handler method.
<i>afterdisconnectedfct</i>	Optional function called just after disconnection of the old datasource.
<i>beforeconnectedfct</i>	Optional function called just before connecting the new datasource.
<i>afterconnectedfct</i>	Optional function called just after connecting the new datasource.

Used internally in widget implementations.

Parameters

<i>widget</i>	The clove::Widget .
<i>datasource</i>	The clove::Datasource to connect.
<i>dspropname</i>	The name of the datasource property.
<i>rootptrpropname</i>	The name of the root value pointer property.
<i>inserthandlername</i>	The name of the insert handler.
<i>removehandlername</i>	The name of the remove handler.
<i>updatehandlername</i>	The name of the update handler.
<i>afterdisconnectedfct</i>	This function is called after the last datasource is disconnected.
<i>beforeconnectedfct</i>	This function is called just before the datasource is connected.
<i>afterconnectedfct</i>	This function is called right after the datasource is connected.

1.91.2.9 `cssLengthToPixels()`

```
cssLengthToPixels (
    v,
    axis )
```

Translates a css length specification string to a numeric pixel value.

Note: This does not make sense for all kinds of lengths, e.g. percentages will not work.

Parameters

<i>v</i>	The css length.
<i>axis</i>	The axis ('x' or 'y').

1.91.2.10 `deferLoading()`

```
deferLoading (
    eload )
```

Defers loading of Clove by an event.

Used only internally and in exotic situations.

Call this before loading is finished (i.e. typically while the document itself loads) and before the `eload` event triggers (i.e. not when it already might have triggered).

See also [runOnLoaded\(\)](#).

Parameters

<i>eload</i>	The clove::Event object which is triggered when it's ready for continue loading.
--------------	--

1.91.2.11 findWidgetForDomNode()

```
findWidgetForDomNode (
    node )
```

Finds the (most inner) `clove::Widget` a given dom node contains to.

Parameters

<i>node</i>	The dom node.
-------------	---------------

1.91.2.12 getExtraSize()

```
getExtraSize (
    node,
    withMargin )
```

Computes the extra size of a dom node.

This is the difference between inner size and the outer size (i.e. borders, padding, ...).

Parameters

<i>node</i>	The dom node to measure.
<i>withMargin</i>	If to include margins.

1.91.2.13 getFullPathForLoadedResource()

```
getFullPathForLoadedResource (
    tgtname,
    elementtype,
    elementsrcattrname )
```

Returns the full path for an already loaded script or stylesheet with known file name.

Used only internally and in exotic situations.

Parameters

<i>tgtname</i>	The complete file name (bare; without slashes) of the file to search for.
<i>elementtype</i>	Optional element type (default: "script").
<i>elementsrcattrname</i>	Optional element type's attribute name for the source url (if type is not "script" or "style").

1.91.2.14 getInnerSize()

```
getInnerSize (
    node )
```

Computes the inner size of a dom node.

This is the size without borders, padding, ...

Parameters

<i>node</i>	The dom node to measure.
-------------	--------------------------

1.91.2.15 getOuterSize()

```
getOuterSize (
    node,
    withMargin )
```

Computes the outer size of a dom node.

This is the size including borders, padding, ...

Parameters

<i>node</i>	The dom node to measure.
<i>withMargin</i>	If to include margins.

1.91.2.16 insertToArray()

```
insertToArray (
    array,
    i,
    v )
```

Inserts a value into an array.

Parameters

<i>array</i>	The array.
<i>i</i>	The position.
<i>v</i>	The new value to insert.

1.91.2.17 isDescendantOf()

```
isDescendantOf (
    d,
    p )
```

Checks if *d* is a descendant of *p* in the html dom node.

Parameters

<i>d</i>	An html dom node.
<i>p</i>	An html dom node.

1.91.2.18 main_parts_resized()

```
main_parts_resized (
    reason )
```

Call this method if resizing is required due to external situations.

It is not required to call this method in typical situations. A situation which requires this method is changing the stylesheets from outside.

Parameters

<i>reason</i>	Optional arbitrary object which describes the reason for resizing. Can be evaluated by handlers in a custom way.
---------------	--

1.91.2.19 OnMainResized()

```
OnMainResized
```

Triggered when the main view resized.

This is a [clove.Event](#) instance. See Manual for details.

1.91.2.20 popup()

```
popup (
    config,
    showconfig )
```

Shows a popup.

This can show an arbitrary widget decoupled from the main user interface (swimming above it).

This call returns a dialog result structure, containing the following:

- `widget`: The popup widget as [clove::Widget](#).
- `namespace`: The root namespace for the popup as [clove::NameScope](#).

Parameters

<i>config</i>	The widget configuration, as for clove::build() , which specifies the popup.
<i>showconfig</i>	A configuration object which specifies some presentation aspects.

1.91.2.21 removeOnDoubleClickTapHandler()

```
removeOnDoubleClickTapHandler (
    domnode,
    handler )
```

Remove a double click/tap event handler.

Parameters

<i>domnode</i>	The node in the dom tree to remove a handler from.
<i>handler</i>	The function to remove.

1.91.2.22 resumeResizing()

```
resumeResizing (
    token )
```

Resumes the user interface resizing after it was suspended with [clove::utils::suspendResizing](#).

It automatically takes care of resizing everything which diverged meanwhile.

Parameters

<i>token</i>	The token object.
--------------	-------------------

1.91.2.23 runOnLoaded()

```
runOnLoaded (
    fct )
```

Runs a function as soon as possible, but not before Clove has finished loading.

Used only internally and in exotic situations. One would typically use [clove::populateUI\(\)](#) instead.

Parameters

<i>fct</i>	The <code>function()</code> to execute when Clove is loaded.
------------	--

1.91.2.24 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    node,
    class,
    assigned )
```

Sets or unsets a css class to an html dom node.

Parameters

<i>node</i>	The html dom node to modify.
<i>class</i>	The css class name.
<i>assigned</i>	If to assign or unassign it.

1.91.2.25 `suspendResizing()`

```
suspendResizing ( )
```

Temporarily suspends all user interface resizing.

This is useful for performance reasons during intensive changes to the user interface.

Use this with caution and always call `clove::utils::resumeResizing` afterwards.

Returns

A token object used for resuming later on.

1.91.2.26 `tryLoadScript()`

```
tryLoadScript (
    scriptpath,
    config )
```

Tries to load another javascript.

If it succeeds, it returns the url of it.

config may contain:

- `rootrefscript`: For relative script paths, this reference script is used in order to determine the base directory.
- `onload`: A `function()` to execute when the script is loaded.

Parameters

<i>scriptpath</i>	The path to the javascript file.
<i>config</i>	A (optional) configuration object.

1.91.2.27 tryLoadStyle()

```
tryLoadStyle (
    stylepath,
    config )
```

Tries to load a css stylesheet.

If it succeeds, it returns the url of it.

config may contain:

- `rootrefscript`: For relative style paths, this reference script is used in order to determine the base directory.
- `onload`: A `function()` to execute when the stylesheet is loaded.

Parameters

<i>stylepath</i>	The path to the css stylesheet file.
<i>config</i>	A (optional) configuration object.

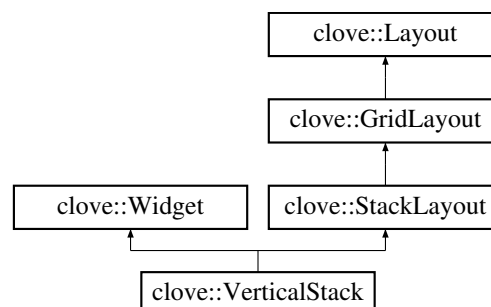
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.92 clove::VerticalStack Class Reference

A container which stacks child widgets row-wise.

Inheritance diagram for clove::VerticalStack:



Public Member Functions

- [rows](#) ()
The list of child widgets as widget configurations like for [clove::build\(\)](#).
- [setRows](#) (value)
Setter for [rows\(\)](#).
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)
Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)
Removes the css class `css` from this widget.
- [isStyleClass](#) (css)

- Checks if this widget contains the css class `class`.*

 - [setStyleClassAssigned](#) (`class`, `assigned`)

Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.
- [nameScope](#) ()

The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (`removeconfig`)

Removes this widget.
- [effectivelyEnabled](#) ()

If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()

If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()

List of the children `clove::Widget` instances.
- [parentWidget](#) ()

The parent `clove::Widget`.
- [name](#) ()

The name of the widget.
- [setName](#) (`value`)

Setter for [name\(\)](#).
- [enabled](#) ()

If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (`value`)

Setter for [enabled\(\)](#).
- [styleClass](#) ()

Custom css class(es).
- [setStyleClass](#) (`value`)

Setter for [styleClass\(\)](#).
- [style](#) ()

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (`value`)

Setter for [style\(\)](#).
- [visibility](#) ()

If this widget is visible.
- [setVisibility](#) (`value`)

Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()

If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (`value`)

Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()

If the widget can have the keyboard focus.
- [setMayFocus](#) (`value`)

Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (`value`)

Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

- [setVerticalStretchAffinity](#) (value)
Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.
- [insertRow](#) (i)
Inserts a new empty row to the grid.
- [insertColumn](#) (i)
Inserts a new empty column to the grid.
- [removeRow](#) (i)

- Removes a row from the grid.*
 - `removeColumn` (*i*)
 - Removes a column from the grid.*
 - `children` ()
 - List of all child widgets in this layout as widget configurations like in `clove::build()`.*
 - `setChildren` (value)
 - Setter for `children()`.*
 - `addChild` (value)
 - Adds a new child widget to the layout.*
 - `clearChilds` ()
 - Removes all childs from the layout.*

1.92.1 Detailed Description

A container which stacks child widgets row-wise.

1.92.2 Member Function Documentation

1.92.2.1 `addChild()`

```
addChild (
    value ) [inherited]
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for <code>clove::build()</code> .
--------------	---

1.92.2.2 `addStyleClass()`

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.92.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.92.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.92.2.5 children()

```
children ( ) [inherited]
```

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.92.2.6 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.92.2.7 clearChilds()

```
clearChilds ( ) [inherited]
```

Removes all childs from the layout.

1.92.2.8 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.92.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.92.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.92.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.92.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.92.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.92.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.92.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.92.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.92.2.17 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.92.2.18 effectivelyEnabled()

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.92.2.19 effectiveVisibility()

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.92.2.20 enabled()

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.92.2.21 focus()

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.92.2.22 getMinimalHeightForWidth()

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.92.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.92.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.92.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.92.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.92.2.27 hasFocus()

hasFocus () [inherited]

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.92.2.28 horizontalStretchAffinity()

horizontalStretchAffinity () [inherited]

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.92.2.29 hstretch()

hstretch () [inherited]

Alias for [horizontalStretchAffinity\(\)](#).

1.92.2.30 init()

init (
 rootNameScope) [inherited]

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.92.2.31 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.92.2.32 insertColumn()

```
insertColumn (
    i ) [inherited]
```

Inserts a new empty column to the grid.

Parameters

<i>i</i>	The column index.
----------	-------------------

1.92.2.33 insertRow()

```
insertRow (
    i ) [inherited]
```

Inserts a new empty row to the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.92.2.34 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.92.2.35 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>css</i>	A css class name.
------------	-------------------

1.92.2.36 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.92.2.37 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.92.2.38 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.92.2.39 OnDestroyed()

```
OnDestroyed [inherited]
```

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.92.2.40 OnKeyDown()

```
OnKeyDown [inherited]
```

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.92.2.41 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.92.2.42 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.92.2.43 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.92.2.44 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.92.2.45 OnVisibilityChanged()

`OnVisibilityChanged` [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.92.2.46 `outerSize()`

`outerSize () [inherited]`

Returns outer width and height of this widget.

1.92.2.47 `parentWidget()`

`parentWidget () [inherited]`

The parent [clove::Widget](#).

1.92.2.48 `registerBusy()`

`registerBusy () [inherited]`

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.92.2.49 `relayout()`

`relayout () [inherited]`

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.92.2.50 `remove()`

`remove (
 removeconfig) [inherited]`

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.92.2.51 removeColumn()

```
removeColumn (
    i ) [inherited]
```

Removes a column from the grid.

Parameters

<i>i</i>	The column index.
----------	-------------------

1.92.2.52 removeRow()

```
removeRow (
    i ) [inherited]
```

Removes a row from the grid.

Parameters

<i>i</i>	The row index.
----------	----------------

1.92.2.53 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.92.2.54 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.92.2.55 `rows()`

```
rows ( )
```

The list of child widgets as widget configurations like for [clove::build\(\)](#).

1.92.2.56 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.57 `setChildren()`

```
setChildren (
    value ) [inherited]
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.58 `setDoStandaloneResizing()`

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.59 `setEnabled()`

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.60 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.61 `setHstretch()`

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.62 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.63 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.64 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.92.2.65 setRows()

```
setRows (
    value )
```

Setter for [rows\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.66 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.67 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.68 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.69 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.70 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.92.2.71 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.72 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.73 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.92.2.74 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.92.2.75 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.92.2.76 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.92.2.77 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.92.2.78 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.92.2.79 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.92.2.80 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.92.2.81 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

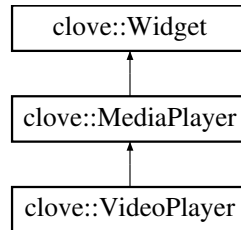
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.93 clove::VideoPlayer Class Reference

A widget which plays a video source and optionally allows user controls.

Inheritance diagram for clove::VideoPlayer:



Public Member Functions

- static [canPlayType](#) (t)
Determines if the browser can play a certain video type and returns a string as described in the html specs (-> "canPlayType").
- [autoPlay](#) ()
If to automatically start playback as soon as possible.
- [setAutoPlay](#) (value)
Setter for [autoPlay\(\)](#).
- [showControls](#) ()
If to show user controls for play, pause and more.
- [setShowControls](#) (value)
Setter for [showControls\(\)](#).
- [currentMediaPosition](#) ()
The current media playback position in seconds.
- [setCurrentMediaPosition](#) (value)
Setter for [currentMediaPosition\(\)](#).
- [loop](#) ()
If to playback in an endless loop.
- [setLoop](#) (value)
Setter for [loop\(\)](#).
- [muted](#) ()
If to mute the audio playback.
- [setMuted](#) (value)
Setter for [muted\(\)](#).
- [preload](#) ()
If to begin loading the media data as soon as possible.
- [setPreload](#) (value)
Setter for [preload\(\)](#).
- [volume](#) ()
The audio playback volume between 0 . 0 and 1 . 0.
- [setVolume](#) (value)
Setter for [volume\(\)](#).
- [source](#) ()
The media source URL.
- [setSource](#) (value)

- Setter for [source\(\)](#).
- [duration](#) ()
 - The duration of the loaded media source in seconds.
- [hasEnded](#) ()
 - If the playback has ended (i.e. reached the end).
- [isPaused](#) ()
 - If the playback is logically paused.
- [nativeNode](#) ()
 - Returns the native html dom node.
- [play](#) ()
 - Starts playback.
- [pause](#) ()
 - Pauses playback.
- [OnLoadingAborted](#)
 - Triggered when the media loading aborted for some reasons (e.g. network errors).
- [OnReadyForPlayback](#)
 - Triggered when there are enough data for starting playback.
- [OnDurationChanged](#)
 - Triggered when the playback duration changed.
- [OnHasEnded](#)
 - Triggered when the playback has ended.
- [OnIsPaused](#)
 - Triggered when the playback logically paused.
- [OnIsPlaying](#)
 - Triggered when the playback logically starts.
- [declareProperty](#) (k, defaultV)
 - Declares a widget property.
- [getProperty](#) (k)
 - General-purpose getter for widget properties.
- [setProperty](#) (k, v)
 - General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
 - Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
 - Initializes the widget.
- [doinit](#) ()
 - Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
 - Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
 - Applies the new widget size to internal content.
- [doresize](#) ()
 - Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
 - Notifies the parent widget that a new geometry is required.
- [focus](#) ()
 - Sets the focus to this widget.
- [isAlive](#) ()
 - Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
 - Computes the minimal width in pixel this widget needs to have.

- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (css)
Adds the css class `css` to this widget.
- [removeStyleClass](#) (css)
Removes the css class `css` from this widget.
- [isStyleClass](#) (css)
Checks if this widget contains the css class `css`.
- [setStyleClassAssigned](#) (css, assigned)
Sets or unsets the css class `css` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)
Setter for [name\(\)](#).
- [enabled](#) ()
If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
Setter for [enabled\(\)](#).
- [styleClass](#) ()
Custom css class(es).
- [setStyleClass](#) (value)
Setter for [styleClass\(\)](#).
- [style](#) ()
Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)

- Setter for `style()`.
- `visibility ()`
 - If this widget is visible.
- `setVisibility (value)`
 - Setter for `visibility()`.
- `doStandaloneResizing ()`
 - If the widget handles to resize itself as needed.
- `setDoStandaloneResizing (value)`
 - Setter for `doStandaloneResizing()`.
- `mayFocus ()`
 - If the widget can have the keyboard focus.
- `setMayFocus (value)`
 - Setter for `mayFocus()`.
- `horizontalStretchAffinity ()`
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- `setHorizontalStretchAffinity (value)`
 - Setter for `horizontalStretchAffinity()`.
- `verticalStretchAffinity ()`
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- `setVerticalStretchAffinity (value)`
 - Setter for `verticalStretchAffinity()`.
- `strictHorizontalSizing ()`
 - If the widget is strictly forbidden to get additional horizontal space.
- `setStrictHorizontalSizing (value)`
 - Setter for `strictHorizontalSizing()`.
- `strictVerticalSizing ()`
 - If the widget is strictly forbidden to get additional vertical space.
- `setStrictVerticalSizing (value)`
 - Setter for `strictVerticalSizing()`.
- `vstretch ()`
 - Alias for `verticalStretchAffinity()`.
- `setVstretch (value)`
 - Setter for `vstretch()`.
- `hstretch ()`
 - Alias for `horizontalStretchAffinity()`.
- `setHstretch (value)`
 - Setter for `hstretch()`.
- `busy ()`
 - If the widget is in busy state, typically resulting in a loading animation.
- `setBusy (value)`
 - Setter for `busy()`.
- `registerBusy ()`
 - Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy (token)`
 - Drops a token and returns to normal state if no other tokens exist.
- `hasFocus ()`
 - If the widget has the keyboard focus.
- `innerSize ()`
 - Returns inner width and height of this widget.
- `outerSize ()`
 - Returns outer width and height of this widget.

- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.

1.93.1 Detailed Description

A widget which plays a video source and optionally allows user controls.

1.93.2 Member Function Documentation

1.93.2.1 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.93.2.2 autoPlay()

```
autoPlay ( ) [inherited]
```

If to automatically start playback as soon as possible.

1.93.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.93.2.4 busy()

```
busy ( ) [inherited]
```

If the widget is in busy state, typically resulting in a loading animation.

1.93.2.5 canPlayType()

```
static canPlayType (
    t )
```

Determines if the browser can play a certain video type and returns a string as described in the html specs (->"canPlayType").

Parameters

<i>t</i>	A mimetype string.
----------	--------------------

1.93.2.6 childrenWidgets()

```
childrenWidgets ( ) [inherited]
```

List of the children [clove::Widget](#) instances.

1.93.2.7 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w ) [inherited]
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.93.2.8 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.93.2.9 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.93.2.10 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.93.2.11 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.93.2.12 currentMediaPosition()

```
currentMediaPosition ( ) [inherited]
```

The current media playback position in seconds.

1.93.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.93.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.93.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.93.2.16 `doresize()`

`doresize () [inherited]`

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.93.2.17 `doStandaloneResizing()`

`doStandaloneResizing () [inherited]`

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.93.2.18 `duration()`

`duration () [inherited]`

The duration of the loaded media source in seconds.

1.93.2.19 `effectivelyEnabled()`

`effectivelyEnabled () [inherited]`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.93.2.20 `effectiveVisibility()`

`effectiveVisibility () [inherited]`

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.93.2.21 `enabled()`

`enabled () [inherited]`

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.93.2.22 focus()

`focus () [inherited]`

Sets the focus to this widget.

1.93.2.23 getMinimalHeightForWidth()

`getMinimalHeightForWidth (
 w) [inherited]`

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.93.2.24 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.93.2.25 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.93.2.26 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.93.2.27 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.93.2.28 hasEnded()

```
hasEnded ( ) [inherited]
```

If the playback has ended (i.e. reached the end).

1.93.2.29 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.93.2.30 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.93.2.31 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.93.2.32 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.93.2.33 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.93.2.34 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.93.2.35 isPaused()

```
isPaused ( ) [inherited]
```

If the playback is logically paused.

This does not return `true` just when loading stalls.

1.93.2.36 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.93.2.37 loop()

```
loop ( ) [inherited]
```


If to playback in an endless loop.

1.93.2.38 mayFocus()

`mayFocus () [inherited]`

If the widget can have the keyboard focus.

1.93.2.39 muted()

`muted () [inherited]`

If to mute the audio playback.

1.93.2.40 name()

`name () [inherited]`

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.93.2.41 nameScope()

`nameScope () [inherited]`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.93.2.42 nativeNode()

`nativeNode () [inherited]`

Returns the native html dom node.

1.93.2.43 OnDestroyed()

`OnDestroyed [inherited]`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.44 OnDurationChanged()

OnDurationChanged [inherited]

Triggered when the playback duration changed.

See also [duration\(\)](#).

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.45 OnHasEnded()

OnHasEnded [inherited]

Triggered when the playback has ended.

See also [hasEnded\(\)](#).

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.46 OnIsPaused()

OnIsPaused [inherited]

Triggered when the playback logically paused.

This is not triggered when loading stalls.

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.47 OnIsPlaying()

OnIsPlaying [inherited]

Triggered when the playback logically starts.

This is not triggered when loading has stalled and resumes.

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.48 OnKeyDown()

OnKeyDown [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.49 OnKeyPress()

OnKeyPress [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.50 OnKeyUp()

OnKeyUp [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.51 OnLoadingAborted()

OnLoadingAborted [inherited]

Triggered when the media loading aborted for some reasons (e.g. network errors).

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.52 OnPropertyChanged()

OnPropertyChanged [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.53 OnReadyForPlayback()

OnReadyForPlayback [inherited]

Triggered when there are enough data for starting playback.

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.54 OnResized()

OnResized [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.55 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.93.2.56 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.93.2.57 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.93.2.58 pause()

pause () [inherited]

Pauses playback.

1.93.2.59 play()

play () [inherited]

Starts playback.

1.93.2.60 preload()

preload () [inherited]

If to begin loading the media data as soon as possible.

1.93.2.61 registerBusy()

```
registerBusy ( ) [inherited]
```

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.93.2.62 relayout()

```
relayout ( ) [inherited]
```

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.93.2.63 remove()

```
remove (
    removeconfig ) [inherited]
```

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.93.2.64 removeStyleClass()

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.93.2.65 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.93.2.66 `setAutoPlay()`

```
setAutoPlay (
    value ) [inherited]
```

Setter for [autoPlay\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.67 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.68 `setCurrentMediaPosition()`

```
setCurrentMediaPosition (
    value ) [inherited]
```

Setter for [currentMediaPosition\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.69 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.70 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.71 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.72 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.73 setLoop()

```
setLoop (
    value ) [inherited]
```

Setter for [loop\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.74 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.75 setMuted()

```
setMuted (
    value ) [inherited]
```

Setter for [muted\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.76 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.77 setPreload()

```
setPreload (
    value ) [inherited]
```

Setter for [preload\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.78 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.93.2.79 setShowControls()

```
setShowControls (
    value ) [inherited]
```

Setter for [showControls\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.80 setSource()

```
setSource (
    value ) [inherited]
```

Setter for [source\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.81 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.82 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.83 `setStyle()`

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.84 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.85 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.93.2.86 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.87 setVisibility()

```
setVisibility (  
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.88 setVolume()

```
setVolume (  
    value ) [inherited]
```

Setter for [volume\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.89 setVstretch()

```
setVstretch (  
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.93.2.90 showControls()

`showControls () [inherited]`

If to show user controls for play, pause and more.

1.93.2.91 source()

`source () [inherited]`

The media source URL.

1.93.2.92 strictHorizontalSizing()

`strictHorizontalSizing () [inherited]`

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.93.2.93 strictVerticalSizing()

`strictVerticalSizing () [inherited]`

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.93.2.94 style()

`style () [inherited]`

Custom css style string. You should not use that, but `styleClass()` instead.

1.93.2.95 styleClass()

`styleClass () [inherited]`

Custom css class(es).

1.93.2.96 unregisterBusy()

`unregisterBusy (
 token) [inherited]`

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.93.2.97 verticalStretchAffinity()

`verticalStretchAffinity () [inherited]`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.93.2.98 visibility()

`visibility () [inherited]`

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.93.2.99 volume()

`volume () [inherited]`

The audio playback volume between 0.0 and 1.0.

1.93.2.100 vstretch()

`vstretch () [inherited]`

Alias for [verticalStretchAffinity\(\)](#).

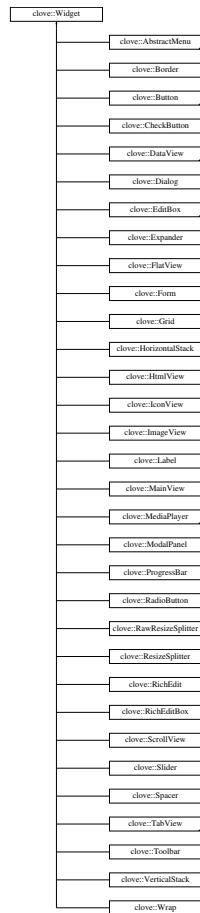
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.94 clove::Widget Class Reference

Base class for a Clove widget.

Inheritance diagram for clove::Widget:



Public Member Functions

- [Widget](#) (config, domnode)
- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).

- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addClass](#) (class)
Adds the css class `class` to this widget.
- [removeClass](#) (class)
Removes the css class `class` from this widget.
- [hasClass](#) (class)
Checks if this widget contains the css class `class`.
- [setStyleClassAssigned](#) (class, assigned)
Sets or unsets the css class `class` to this widget.
- [containingNameScope](#) ()
The namespace this widget contains to.
- [nameScope](#) ()
The own namespace (or the namespace it contains to, if this widget does not bring a new one).
- [remove](#) (removeconfig)
Removes this widget.
- [effectivelyEnabled](#) ()
If this widget is enabled and not marked as busy (i.e. can interact with the user).
- [effectiveVisibility](#) ()
If this widget and all parent widgets are visible. See also [visibility\(\)](#).
- [childrenWidgets](#) ()
List of the children `clove::Widget` instances.
- [parentWidget](#) ()
The parent `clove::Widget`.
- [name](#) ()
The name of the widget.
- [setName](#) (value)

- Setter for [name\(\)](#).
- [enabled](#) ()
 - If this widget is marked as enabled (i.e. can interact with the user).
- [setEnabled](#) (value)
 - Setter for [enabled\(\)](#).
- [styleClass](#) ()
 - Custom css class(es).
- [setStyleClass](#) (value)
 - Setter for [styleClass\(\)](#).
- [style](#) ()
 - Custom css style string. You should not use that, but [styleClass\(\)](#) instead.
- [setStyle](#) (value)
 - Setter for [style\(\)](#).
- [visibility](#) ()
 - If this widget is visible.
- [setVisibility](#) (value)
 - Setter for [visibility\(\)](#).
- [doStandaloneResizing](#) ()
 - If the widget handles to resize itself as needed.
- [setDoStandaloneResizing](#) (value)
 - Setter for [doStandaloneResizing\(\)](#).
- [mayFocus](#) ()
 - If the widget can have the keyboard focus.
- [setMayFocus](#) (value)
 - Setter for [mayFocus\(\)](#).
- [horizontalStretchAffinity](#) ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.
- [setHorizontalStretchAffinity](#) (value)
 - Setter for [horizontalStretchAffinity\(\)](#).
- [verticalStretchAffinity](#) ()
 - Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.
- [setVerticalStretchAffinity](#) (value)
 - Setter for [verticalStretchAffinity\(\)](#).
- [strictHorizontalSizing](#) ()
 - If the widget is strictly forbidden to get additional horizontal space.
- [setStrictHorizontalSizing](#) (value)
 - Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
 - If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
 - Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
 - Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
 - Setter for [vstretch\(\)](#).
- [hstretch](#) ()
 - Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
 - Setter for [hstretch\(\)](#).
- [busy](#) ()
 - If the widget is in busy state, typically resulting in a loading animation.

- `setBusy` (value)
Setter for `busy()`.
- `registerBusy` ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- `unregisterBusy` (token)
Drops a token and returns to normal state if no other tokens exist.
- `hasFocus` ()
If the widget has the keyboard focus.
- `innerSize` ()
Returns inner width and height of this widget.
- `outerSize` ()
Returns outer width and height of this widget.
- `OnDestroyed`
Triggered when this widget was removed somehow.
- `OnResized`
Triggered when this widget was resized.
- `OnVisibilityChanged`
Triggered when the visibility of this widget changed.
- `OnPropertyChanged`
Triggered whenever a property of this widget changed its value.
- `OnKeyDown`
Triggered when a keyboard key went down.
- `OnKeyUp`
Triggered when a keyboard key came up.
- `OnKeyPress`
Triggered when a keyboard key was pressed.

1.94.1 Detailed Description

Base class for a Clove widget.

Read the Manual about widgets.

1.94.2 Constructor & Destructor Documentation

1.94.2.1 `Widget()`

```
Widget (
    config,
    domnode )
```

Note: All widgets have a constructor with this signature. The configuration parameters are stored in `config`.

It is typically not required to call those constructors directly. Use `clove::build()` or some other factory functions (for special situations) instead.

Parameters

<i>config</i>	A widget configuration as in clove::build() .
<i>domnode</i>	The dom node to use as host for this widget.

1.94.3 Member Function Documentation

1.94.3.1 addStyleClass()

```
addStyleClass (
    class )
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.94.3.2 bindProperty()

```
bindProperty (
    k,
    vb )
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.94.3.3 busy()

```
busy ( )
```

If the widget is in busy state, typically resulting in a loading animation.

1.94.3.4 childrenWidgets()

```
childrenWidgets ( )
```

List of the children [Clove::Widget](#) instances.

1.94.3.5 computeMinimalHeightForWidth()

```
computeMinimalHeightForWidth (
    w )
```

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.94.3.6 computeMinimalWidth()

```
computeMinimalWidth ( )
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.94.3.7 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w )
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.94.3.8 computePreferredWidth()

```
computePreferredWidth ( )
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.94.3.9 containingNameScope()

```
containingNameScope ( )
```

The namespace this widget contains to.

1.94.3.10 declareProperty()

```
declareProperty (
    k,
    defaultV )
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.94.3.11 doinit()

```
doinit ( )
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.94.3.12 doinitEarly()

```
doinitEarly ( )
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.94.3.13 `doresize()`

```
doresize ( )
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.94.3.14 `doStandaloneResizing()`

```
doStandaloneResizing ( )
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.94.3.15 `effectivelyEnabled()`

```
effectivelyEnabled ( )
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.94.3.16 `effectiveVisibility()`

```
effectiveVisibility ( )
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.94.3.17 `enabled()`

```
enabled ( )
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.94.3.18 `focus()`

```
focus ( )
```

Sets the focus to this widget.

1.94.3.19 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w )
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.94.3.20 getMinimalWidth()

```
getMinimalWidth ( )
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.94.3.21 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w )
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.94.3.22 getPreferredWidth()

```
getPreferredWidth ( )
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.94.3.23 getProperty()

```
getProperty (
    k )
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty("name")` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.94.3.24 hasFocus()

```
hasFocus ( )
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.94.3.25 horizontalStretchAffinity()

```
horizontalStretchAffinity ( )
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.94.3.26 hstretch()

```
hstretch ( )
```

Alias for [horizontalStretchAffinity\(\)](#).

1.94.3.27 init()

```
init (
    rootNameScope )
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.94.3.28 innerSize()

```
innerSize ( )
```

Returns inner width and height of this widget.

1.94.3.29 isAlive()

```
isAlive ( )
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.94.3.30 isStyleClass()

```
isStyleClass (
    class )
```

Checks if this widget contains the css class `class`.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.94.3.31 mayFocus()

```
mayFocus ( )
```

If the widget can have the keyboard focus.

1.94.3.32 name()

```
name ( )
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.94.3.33 nameScope()

`nameScope ()`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.94.3.34 OnDestroyed()

`OnDestroyed`

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.94.3.35 OnKeyDown()

`OnKeyDown`

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.94.3.36 OnKeyPress()

`OnKeyPress`

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.94.3.37 OnKeyUp()

`OnKeyUp`

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.94.3.38 OnPropertyChanged()

`OnPropertyChanged`

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.94.3.39 OnResized()

`OnResized`

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.94.3.40 OnVisibilityChanged()

`OnVisibilityChanged`

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.94.3.41 outerSize()

`outerSize ()`

Returns outer width and height of this widget.

1.94.3.42 parentWidget()

`parentWidget ()`

The parent [clove::Widget](#).

1.94.3.43 registerBusy()

`registerBusy ()`

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.94.3.44 relayout()

`relayout ()`

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.94.3.45 remove()

`remove (
 removeconfig)`

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.94.3.46 removeStyleClass()

```
removeStyleClass (
    class )
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.94.3.47 resize()

```
resize ( )
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.94.3.48 setBusy()

```
setBusy (
    value )
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.49 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value )
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.50 `setEnabled()`

```
setEnabled (
    value )
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.51 `setHorizontalStretchAffinity()`

```
setHorizontalStretchAffinity (
    value )
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.52 `setHstretch()`

```
setHstretch (
    value )
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.53 setMayFocus()

```
setMayFocus (
    value )
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.54 setName()

```
setName (
    value )
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.55 setProperty()

```
setProperty (
    k,
    v )
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.94.3.56 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value )
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.57 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value )
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.58 setStyle()

```
setStyle (
    value )
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.59 setStyleClass()

```
setStyleClass (
    value )
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.60 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned )
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.94.3.61 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value )
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.62 `setVisibility()`

```
setVisibility (
    value )
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.63 `setVstretch()`

```
setVstretch (
    value )
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.94.3.64 strictHorizontalSizing()

```
strictHorizontalSizing ( )
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with `horizontalStretchAffinity() == 0`.

1.94.3.65 strictVerticalSizing()

```
strictVerticalSizing ( )
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with `verticalStretchAffinity() == 0`.

1.94.3.66 style()

```
style ( )
```

Custom css style string. You should not use that, but `styleClass()` instead.

1.94.3.67 styleClass()

```
styleClass ( )
```

Custom css class(es).

1.94.3.68 unregisterBusy()

```
unregisterBusy (
    token )
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.94.3.69 `verticalStretchAffinity()`

```
verticalStretchAffinity ( )
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.94.3.70 `visibility()`

```
visibility ( )
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.94.3.71 `vstretch()`

```
vstretch ( )
```

Alias for [verticalStretchAffinity\(\)](#).

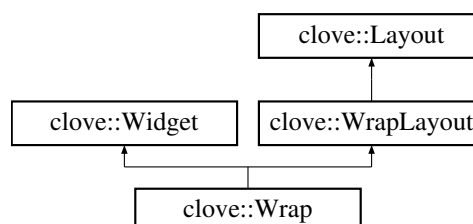
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.95 `clove::Wrap` Class Reference

A container for aligning child widgets in horizontal wrapping lines.

Inheritance diagram for `clove::Wrap`:



Public Member Functions

- [declareProperty](#) (k, defaultV)
Declares a widget property.
- [getProperty](#) (k)
General-purpose getter for widget properties.
- [setProperty](#) (k, v)
General-purpose setter for widget properties.
- [bindProperty](#) (k, vb)
Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.
- [init](#) (rootNameScope)
Initializes the widget.
- [doinit](#) ()
Executes late widget initialization (i.e. after properties are applied).
- [doinitEarly](#) ()
Executes early widget initialization (i.e. before properties are applied).
- [resize](#) ()
Applies the new widget size to internal content.
- [doresize](#) ()
Corrects alignments of internal elements according to the new widget size.
- [relayout](#) ()
Notifies the parent widget that a new geometry is required.
- [focus](#) ()
Sets the focus to this widget.
- [isAlive](#) ()
Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).
- [computeMinimalWidth](#) ()
Computes the minimal width in pixel this widget needs to have.
- [computePreferredWidth](#) ()
Computes the preferred width in pixel this widget needs to have.
- [computeMinimalHeightForWidth](#) (w)
Computes the minimal height in pixel this widget needs to have for a given width.
- [computePreferredHeightForWidth](#) (w)
Computes the preferred height in pixel this widget needs to have for a given width.
- [getMinimalWidth](#) ()
Returns the minimal width in pixel this widget needs to have.
- [getPreferredWidth](#) ()
Returns the preferred width in pixel this widget needs to have.
- [getMinimalHeightForWidth](#) (w)
Returns the minimal height in pixel this widget needs to have for a given width.
- [getPreferredHeightForWidth](#) (w)
Returns the preferred height in pixel this widget needs to have for a given width.
- [addStyleClass](#) (clss)
Adds the css class `clss` to this widget.
- [removeStyleClass](#) (clss)
Removes the css class `clss` from this widget.
- [isStyleClass](#) (clss)
Checks if this widget contains the css class `clss`.
- [setStyleClassAssigned](#) (clss, assigned)
Sets or unsets the css class `clss` to this widget.
- [containingNameScope](#) ()

The namespace this widget contains to.

- `nameScope ()`

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

- `remove (removeconfig)`

Removes this widget.

- `effectivelyEnabled ()`

If this widget is enabled and not marked as busy (i.e. can interact with the user).

- `effectiveVisibility ()`

If this widget and all parent widgets are visible. See also `visibility()`.

- `childrenWidgets ()`

List of the children `clove::Widget` instances.

- `parentWidget ()`

The parent `clove::Widget`.

- `name ()`

The name of the widget.

- `setName (value)`

Setter for `name()`.

- `enabled ()`

If this widget is marked as enabled (i.e. can interact with the user).

- `setEnabled (value)`

Setter for `enabled()`.

- `styleClass ()`

Custom css class(es).

- `setStyleClass (value)`

Setter for `styleClass()`.

- `style ()`

Custom css style string. You should not use that, but `styleClass()` instead.

- `setStyle (value)`

Setter for `style()`.

- `visibility ()`

If this widget is visible.

- `setVisibility (value)`

Setter for `visibility()`.

- `doStandaloneResizing ()`

If the widget handles to resize itself as needed.

- `setDoStandaloneResizing (value)`

Setter for `doStandaloneResizing()`.

- `mayFocus ()`

If the widget can have the keyboard focus.

- `setMayFocus (value)`

Setter for `mayFocus()`.

- `horizontalStretchAffinity ()`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

- `setHorizontalStretchAffinity (value)`

Setter for `horizontalStretchAffinity()`.

- `verticalStretchAffinity ()`

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

- `setVerticalStretchAffinity (value)`

Setter for `verticalStretchAffinity()`.

- `strictHorizontalSizing ()`

If the widget is strictly forbidden to get additional horizontal space.

- [setStrictHorizontalSizing](#) (value)
Setter for [strictHorizontalSizing\(\)](#).
- [strictVerticalSizing](#) ()
If the widget is strictly forbidden to get additional vertical space.
- [setStrictVerticalSizing](#) (value)
Setter for [strictVerticalSizing\(\)](#).
- [vstretch](#) ()
Alias for [verticalStretchAffinity\(\)](#).
- [setVstretch](#) (value)
Setter for [vstretch\(\)](#).
- [hstretch](#) ()
Alias for [horizontalStretchAffinity\(\)](#).
- [setHstretch](#) (value)
Setter for [hstretch\(\)](#).
- [busy](#) ()
If the widget is in busy state, typically resulting in a loading animation.
- [setBusy](#) (value)
Setter for [busy\(\)](#).
- [registerBusy](#) ()
Sets the widget to busy state and returns a token which helps returning to normal state.
- [unregisterBusy](#) (token)
Drops a token and returns to normal state if no other tokens exist.
- [hasFocus](#) ()
If the widget has the keyboard focus.
- [innerSize](#) ()
Returns inner width and height of this widget.
- [outerSize](#) ()
Returns outer width and height of this widget.
- [OnDestroyed](#)
Triggered when this widget was removed somehow.
- [OnResized](#)
Triggered when this widget was resized.
- [OnVisibilityChanged](#)
Triggered when the visibility of this widget changed.
- [OnPropertyChanged](#)
Triggered whenever a property of this widget changed its value.
- [OnKeyDown](#)
Triggered when a keyboard key went down.
- [OnKeyUp](#)
Triggered when a keyboard key came up.
- [OnKeyPress](#)
Triggered when a keyboard key was pressed.
- [children](#) ()
List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).
- [setChildren](#) (value)
Setter for [children\(\)](#).
- [addChild](#) (value)
Adds a new child widget to the layout.
- [clearChilds](#) ()
Removes all childs from the layout.

1.95.1 Detailed Description

A container for aligning child widgets in horizontal wrapping lines.

1.95.2 Member Function Documentation

1.95.2.1 addChild()

```
addChild (
    value ) [inherited]
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for clove::build() .
--------------	--

1.95.2.2 addStyleClass()

```
addStyleClass (
    class ) [inherited]
```

Adds the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.95.2.3 bindProperty()

```
bindProperty (
    k,
    vb ) [inherited]
```

Binds a [DataBinding](#) to a property. Read Manual for details about data bindings.

Parameters

<i>k</i>	The widget property name.
<i>vb</i>	The clove.DataBinding to bind. May be 'undefined' for just unbinding.

1.95.2.4 busy()

`busy () [inherited]`

If the widget is in busy state, typically resulting in a loading animation.

1.95.2.5 children()

`children () [inherited]`

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.95.2.6 childrenWidgets()

`childrenWidgets () [inherited]`

List of the children [clove::Widget](#) instances.

1.95.2.7 clearChilds()

`clearChilds () [inherited]`

Removes all childs from the layout.

1.95.2.8 computeMinimalHeightForWidth()

`computeMinimalHeightForWidth (
 w) [inherited]`

Computes the minimal height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.95.2.9 computeMinimalWidth()

```
computeMinimalWidth ( ) [inherited]
```

Computes the minimal width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.95.2.10 computePreferredHeightForWidth()

```
computePreferredHeightForWidth (
    w ) [inherited]
```

Computes the preferred height in pixel this widget needs to have for a given width.

Override this method for geometry measurement in custom widgets.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.95.2.11 computePreferredWidth()

```
computePreferredWidth ( ) [inherited]
```

Computes the preferred width in pixel this widget needs to have.

Override this method for geometry measurement in custom widgets.

1.95.2.12 containingNameScope()

```
containingNameScope ( ) [inherited]
```

The namespace this widget contains to.

1.95.2.13 declareProperty()

```
declareProperty (
    k,
    defaultV ) [inherited]
```

Declares a widget property.

This is intended to be called only from inside the widget constructor.

Read the Manual about widget properties and custom widgets.

Parameters

<i>k</i>	The widget property name.
<i>defaultV</i>	The default value.

1.95.2.14 doinit()

```
doinit ( ) [inherited]
```

Executes late widget initialization (i.e. after properties are applied).

This is used for initialization steps which need the properties to be applied already. See also [doinitEarly\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.95.2.15 doinitEarly()

```
doinitEarly ( ) [inherited]
```

Executes early widget initialization (i.e. before properties are applied).

This is used for initialization steps which need to run before property values are applied. See also [doinit\(\)](#).

Override this method in custom widgets.

Intended for usage by the infrastructure; never call this method directly.

1.95.2.16 doresize()

```
doresize ( ) [inherited]
```

Corrects alignments of internal elements according to the new widget size.

Override this method in custom widgets.

Never call this method directly. See [resize\(\)](#).

1.95.2.17 doStandaloneResizing()

```
doStandaloneResizing ( ) [inherited]
```

If the widget handles to resize itself as needed.

This is only needed in exotic situations and by the infrastructure.

1.95.2.18 `effectivelyEnabled()`

```
effectivelyEnabled ( ) [inherited]
```

If this widget is enabled and not marked as busy (i.e. can interact with the user).

This checks if the parent widgets are enabled and non-busy as well. See also [enabled\(\)](#) and [busy\(\)](#).

1.95.2.19 `effectiveVisibility()`

```
effectiveVisibility ( ) [inherited]
```

If this widget and all parent widgets are visible. See also [visibility\(\)](#).

1.95.2.20 `enabled()`

```
enabled ( ) [inherited]
```

If this widget is marked as enabled (i.e. can interact with the user).

This does not check the parents. See also [effectivelyEnabled\(\)](#).

1.95.2.21 `focus()`

```
focus ( ) [inherited]
```

Sets the focus to this widget.

1.95.2.22 `getMinimalHeightForWidth()`

```
getMinimalHeightForWidth (
    w ) [inherited]
```

Returns the minimal height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.95.2.23 getMinimalWidth()

```
getMinimalWidth ( ) [inherited]
```

Returns the minimal width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.95.2.24 getPreferredHeightForWidth()

```
getPreferredHeightForWidth (
    w ) [inherited]
```

Returns the preferred height in pixel this widget needs to have for a given width.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

Parameters

<i>w</i>	The width in pixel.
----------	---------------------

1.95.2.25 getPreferredWidth()

```
getPreferredWidth ( ) [inherited]
```

Returns the preferred width in pixel this widget needs to have.

Do not override this method in custom widgets. This method caches geometry and does some other adjustments.

1.95.2.26 getProperty()

```
getProperty (
    k ) [inherited]
```

General-purpose getter for widget properties.

Typically used for handling properties, which do not directly belong to the widget class but which are helpers for external aspects (e.g. a row index for positioning in a grid).

The known ones have a dedicated getter and setter, i.e. `x.name()` for `'x.getProperty('name')` and `x.setName(v)` respectively.

Read the Manual about widget properties.

Parameters

<i>k</i>	The widget property name.
----------	---------------------------

1.95.2.27 hasFocus()

```
hasFocus ( ) [inherited]
```

If the widget has the keyboard focus.

This also returns true if a child has focus.

1.95.2.28 horizontalStretchAffinity()

```
horizontalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional horizontal space.

1.95.2.29 hstretch()

```
hstretch ( ) [inherited]
```

Alias for [horizontalStretchAffinity\(\)](#).

1.95.2.30 init()

```
init (
    rootNameScope ) [inherited]
```

Initializes the widget.

Never override this method in custom widgets. See [doresize\(\)](#).

Intended for usage by the infrastructure; never call this method directly.

Parameters

<i>rootNameScope</i>	The root namespace to add this widget to.
----------------------	---

1.95.2.31 innerSize()

```
innerSize ( ) [inherited]
```

Returns inner width and height of this widget.

1.95.2.32 isAlive()

```
isAlive ( ) [inherited]
```

Checks if the widget is still alive (i.e. mounted somewhere in the dom tree).

1.95.2.33 isStyleClass()

```
isStyleClass (
    class ) [inherited]
```

Checks if this widget contains the css class `class`.

Parameters

<code>class</code>	A css class name.
--------------------	-------------------

1.95.2.34 mayFocus()

```
mayFocus ( ) [inherited]
```

If the widget can have the keyboard focus.

1.95.2.35 name()

```
name ( ) [inherited]
```

The name of the widget.

This name can be used for getting the widget instance later on. Read the Manual about namespaces for details.

1.95.2.36 nameScope()

```
nameScope ( ) [inherited]
```

The own namespace (or the namespace it contains to, if this widget does not bring a new one).

1.95.2.37 OnDestroyed()

`OnDestroyed` [inherited]

Triggered when this widget was removed somehow.

This is a [clove.Event](#) instance. See Manual for details.

1.95.2.38 OnKeyDown()

`OnKeyDown` [inherited]

Triggered when a keyboard key went down.

This is a [clove.Event](#) instance. See Manual for details.

1.95.2.39 OnKeyPress()

`OnKeyPress` [inherited]

Triggered when a keyboard key was pressed.

Note that this event does not work for all keys in all browsers!

This is a [clove.Event](#) instance. See Manual for details.

1.95.2.40 OnKeyUp()

`OnKeyUp` [inherited]

Triggered when a keyboard key came up.

This is a [clove.Event](#) instance. See Manual for details.

1.95.2.41 OnPropertyChanged()

`OnPropertyChanged` [inherited]

Triggered whenever a property of this widget changed its value.

Note: A few properties are only computed on-demand and don't trigger this event.

This is a [clove.Event](#) instance. See Manual for details.

1.95.2.42 OnResized()

`OnResized` [inherited]

Triggered when this widget was resized.

This is a [clove.Event](#) instance. See Manual for details.

1.95.2.43 OnVisibilityChanged()

OnVisibilityChanged [inherited]

Triggered when the visibility of this widget changed.

Only direct changes trigger the event, not when the visibility of a predecessor changed.

This is a [clove.Event](#) instance. See Manual for details.

1.95.2.44 outerSize()

outerSize () [inherited]

Returns outer width and height of this widget.

1.95.2.45 parentWidget()

parentWidget () [inherited]

The parent [clove::Widget](#).

1.95.2.46 registerBusy()

registerBusy () [inherited]

Sets the widget to busy state and returns a token which helps returning to normal state.

See also [unregisterBusy\(\)](#) and [busy\(\)](#).

You must not mix this function with direct usage of [setBusy\(\)](#)!

1.95.2.47 relayout()

relayout () [inherited]

Notifies the parent widget that a new geometry is required.

This will eventually lead to a resizing according to [clove::Widget::getPreferredWidth](#) (et al).

This function is typically called from within the widget implementation when the content changed and from child widgets.

1.95.2.48 remove()

remove (
 removeconfig) [inherited]

Removes this widget.

Note: This method does not guarantee to remove the widget synchronously, since there might be animations before.

Use [clove::Widget::OnDestroyed](#) for continuing after removal.

Parameters

<i>removeconfig</i>	The removal configuration (optional and only for exotic cases).
---------------------	---

1.95.2.49 `removeStyleClass()`

```
removeStyleClass (
    class ) [inherited]
```

Removes the css class `class` from this widget.

Parameters

<i>class</i>	A css class name.
--------------	-------------------

1.95.2.50 `resize()`

```
resize ( ) [inherited]
```

Applies the new widget size to internal content.

Never override this method in custom widgets. See [doresize\(\)](#).

This function is typically called from the parent widget when the size has been changed.

1.95.2.51 `setBusy()`

```
setBusy (
    value ) [inherited]
```

Setter for [busy\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.52 `setChildren()`

```
setChildren (
    value ) [inherited]
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.53 setDoStandaloneResizing()

```
setDoStandaloneResizing (
    value ) [inherited]
```

Setter for [doStandaloneResizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.54 setEnabled()

```
setEnabled (
    value ) [inherited]
```

Setter for [enabled\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.55 setHorizontalStretchAffinity()

```
setHorizontalStretchAffinity (
    value ) [inherited]
```

Setter for [horizontalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.56 setHstretch()

```
setHstretch (
    value ) [inherited]
```

Setter for [hstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.57 setMayFocus()

```
setMayFocus (
    value ) [inherited]
```

Setter for [mayFocus\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.58 setName()

```
setName (
    value ) [inherited]
```

Setter for [name\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.59 setProperty()

```
setProperty (
    k,
    v ) [inherited]
```

General-purpose setter for widget properties.

See [getProperty\(\)](#).

Parameters

<i>k</i>	The widget property name.
<i>v</i>	The new value.

1.95.2.60 setStrictHorizontalSizing()

```
setStrictHorizontalSizing (
    value ) [inherited]
```

Setter for [strictHorizontalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.61 setStrictVerticalSizing()

```
setStrictVerticalSizing (
    value ) [inherited]
```

Setter for [strictVerticalSizing\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.62 setStyle()

```
setStyle (
    value ) [inherited]
```

Setter for [style\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.63 `setStyleClass()`

```
setStyleClass (
    value ) [inherited]
```

Setter for [styleClass\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.64 `setStyleClassAssigned()`

```
setStyleClassAssigned (
    class,
    assigned ) [inherited]
```

Sets or unsets the css class `class` to this widget.

Parameters

<i>class</i>	A css class name.
<i>assigned</i>	If to assign or unassign it.

1.95.2.65 `setVerticalStretchAffinity()`

```
setVerticalStretchAffinity (
    value ) [inherited]
```

Setter for [verticalStretchAffinity\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.66 `setVisibility()`

```
setVisibility (
    value ) [inherited]
```

Setter for [visibility\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.67 setVstretch()

```
setVstretch (
    value ) [inherited]
```

Setter for [vstretch\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

1.95.2.68 strictHorizontalSizing()

```
strictHorizontalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional horizontal space.

Otherwise there are situations where additional space is assigned even with [horizontalStretchAffinity\(\)](#)==0.

1.95.2.69 strictVerticalSizing()

```
strictVerticalSizing ( ) [inherited]
```

If the widget is strictly forbidden to get additional vertical space.

Otherwise there are situations where additional space is assigned even with [verticalStretchAffinity\(\)](#)==0.

1.95.2.70 style()

```
style ( ) [inherited]
```

Custom css style string. You should not use that, but [styleClass\(\)](#) instead.

1.95.2.71 styleClass()

```
styleClass ( ) [inherited]
```

Custom css class(es).

1.95.2.72 unregisterBusy()

```
unregisterBusy (
    token ) [inherited]
```

Drops a token and returns to normal state if no other tokens exist.

Parameters

<i>token</i>	The token as returned by registerBusy() .
--------------	---

1.95.2.73 verticalStretchAffinity()

```
verticalStretchAffinity ( ) [inherited]
```

Numeric value ≥ 0 which specifies how much this widget would benefit from additional vertical space.

1.95.2.74 visibility()

```
visibility ( ) [inherited]
```

If this widget is visible.

One of [clove::Visible](#), [clove::Invisible](#) and [clove::InvisibleCollapsed](#).

See also [effectiveVisibility\(\)](#).

1.95.2.75 vstretch()

```
vstretch ( ) [inherited]
```

Alias for [verticalStretchAffinity\(\)](#).

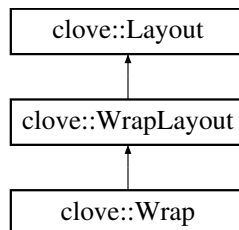
The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

1.96 clove::WrapLayout Class Reference

A mixin for wrap layout implementations.

Inheritance diagram for clove::WrapLayout:



Public Member Functions

- [children](#) ()
List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).
- [setChildren](#) (value)
Setter for [children\(\)](#).
- [addChild](#) (value)
Adds a new child widget to the layout.
- [clearChilds](#) ()
Removes all childs from the layout.

1.96.1 Detailed Description

A mixin for wrap layout implementations.

1.96.2 Member Function Documentation

1.96.2.1 addChild()

```
addChild (
    value ) [inherited]
```

Adds a new child widget to the layout.

Parameters

<i>value</i>	The new widget as widget configuration like for clove::build() .
--------------	--

1.96.2.2 children()

```
children ( ) [inherited]
```

List of all child widgets in this layout as widget configurations like in [clove::build\(\)](#).

1.96.2.3 clearChilds()

```
clearChilds ( ) [inherited]
```

Removes all childs from the layout.

1.96.2.4 setChildren()

```
setChildren (
    value ) [inherited]
```

Setter for [children\(\)](#).

Parameters

<i>value</i>	The new value.
--------------	----------------

The documentation for this class was generated from the following file:

- [_meta/readme/clove.js](#)

Index

actions

- clove::AbstractMenu, [27](#)
- clove::MainView, [601](#)
- clove::Menubar, [653](#)
- clove::PopupMenu, [786](#)
- clove::PopupMenuButton, [807](#)
- clove::Toolbar, [1126](#)

addBodyWidget

- clove::ResizeSplitter, [903](#)

addChild

- clove::FlatLayout, [381](#)
- clove::FlatView, [389](#)
- clove::Grid, [432](#)
- clove::GridLayout, [451](#)
- clove::HorizontalStack, [459](#)
- clove::Layout, [564](#)
- clove::StackLayout, [1041](#)
- clove::VerticalStack, [1191](#)
- clove::Wrap, [1262](#)
- clove::WrapLayout, [1279](#)

addChildNameScope

- clove::NameScope, [715](#)
- clove::RootNameScope, [966](#)

addColumn

- clove::NativeDataSourceNode, [727](#)

addHandler

- clove::Event, [347](#)

addHandlerOnce

- clove::Event, [347](#)

addOnDestroyHandler

- clove::utils, [1177](#)

addOnDoubleClickTapHandler

- clove::utils, [1178](#)

addOnDragHandler

- clove::utils, [1178](#)

addRow

- clove::NativeDataSourceNode, [729](#)

addString

- clove::I18N, [499](#)

addStyleClass

- clove::AbstractMenu, [27](#)
- clove::AudioPlayer, [67](#)
- clove::Border, [93](#)
- clove::Button, [113](#)
- clove::Carousel, [135](#)
- clove::CheckButton, [160](#)
- clove::DataView, [203](#)
- clove::DateBox, [233](#)
- clove::Dialog, [257](#)

- clove::DropDownBox, [279](#)

- clove::EditBox, [303](#)

- clove::EditComboBox, [326](#)

- clove::Expander, [353](#)

- clove::FlatView, [389](#)

- clove::Form, [412](#)

- clove::Grid, [432](#)

- clove::HorizontalStack, [460](#)

- clove::HtmlView, [482](#)

- clove::IconView, [505](#)

- clove::ImageView, [525](#)

- clove::Label, [546](#)

- clove::ListView, [570](#)

- clove::MainView, [601](#)

- clove::MediaPlayer, [628](#)

- clove::Menubar, [653](#)

- clove::ModalPanel, [674](#)

- clove::MultilineEditBox, [695](#)

- clove::NumericEditBox, [738](#)

- clove::PasswordEditBox, [763](#)

- clove::PopupMenu, [786](#)

- clove::PopupMenuButton, [808](#)

- clove::ProgressBar, [830](#)

- clove::RadioButton, [858](#)

- clove::RawResizeSplitter, [883](#)

- clove::ResizeSplitter, [904](#)

- clove::RichEdit, [927](#)

- clove::RichEditBox, [947](#)

- clove::ScrollView, [971](#)

- clove::Slider, [993](#)

- clove::Spacer, [1024](#)

- clove::TabView, [1079](#)

- clove::TableView, [1049](#)

- clove::TimeBox, [1103](#)

- clove::Toolbar, [1127](#)

- clove::TreeView, [1150](#)

- clove::VerticalStack, [1191](#)

- clove::VideoPlayer, [1215](#)

- clove::Widget, [1243](#)

- clove::Wrap, [1262](#)

addTab

- clove::Carousel, [135](#)

- clove::TabView, [1079](#)

AjaxAsyncDataSource

- clove::AjaxAsyncDataSource, [46](#)

allowChecking

- clove::DataView, [203](#)

- clove::ListView, [571](#)

- clove::TableView, [1049](#)

- clove::TreeView, 1150
- allowSelection
 - clove::DataView, 203
 - clove::ListView, 571
 - clove::TableView, 1049
 - clove::TreeView, 1150
- alwaysAllocateExpanderSpace
 - clove::DataView, 203
 - clove::ListView, 571
 - clove::TableView, 1049
 - clove::TreeView, 1150
- animateNode
 - clove::utils, 1178
- appendColumn
 - clove::NativeDatasource, 718
- appendRow
 - clove::NativeDatasource, 718
- applyDefaultsToConfig
 - clove::utils, 1179
- arrayToDatasource
 - clove::utils, 1179
- arraysum
 - clove::utils, 1179
- async_pull
 - clove::AjaxAsyncDatasource, 47
 - clove::AsyncDatasource, 56
- async_push
 - clove::AjaxAsyncDatasource, 47
 - clove::AsyncDatasource, 57
- AsyncDatasource
 - clove::AsyncDatasource, 56
- autoPlay
 - clove::AudioPlayer, 68
 - clove::MediaPlayer, 628
 - clove::VideoPlayer, 1215
- autocompletionFilter
 - clove::DateBox, 233
 - clove::DropDownBox, 279
 - clove::EditBox, 303
 - clove::EditComboBox, 327
 - clove::MultilineEditBox, 695
 - clove::NumericEditBox, 738
 - clove::PasswordEditBox, 763
 - clove::TimeBox, 1104
- autocompletionItems
 - clove::DateBox, 233
 - clove::DropDownBox, 279
 - clove::EditBox, 303
 - clove::EditComboBox, 327
 - clove::MultilineEditBox, 695
 - clove::NumericEditBox, 738
 - clove::PasswordEditBox, 764
 - clove::TimeBox, 1104
- autocompletionOpenForNoText
 - clove::DateBox, 234
 - clove::DropDownBox, 279
 - clove::EditBox, 303
 - clove::EditComboBox, 327
- clove::MultilineEditBox, 695
 - clove::NumericEditBox, 738
 - clove::PasswordEditBox, 764
 - clove::TimeBox, 1104
- backendObject
 - clove::DatasourceValuePointer, 194
- bind
 - clove::DataBinding, 187
- bindProperty
 - clove::AbstractMenu, 28
 - clove::AudioPlayer, 68
 - clove::Border, 93
 - clove::Button, 113
 - clove::Carousel, 136
 - clove::CheckBox, 160
 - clove::DataView, 203
 - clove::DateBox, 234
 - clove::Dialog, 257
 - clove::DropDownBox, 279
 - clove::EditBox, 304
 - clove::EditComboBox, 327
 - clove::Expander, 353
 - clove::FlatView, 390
 - clove::Form, 413
 - clove::Grid, 433
 - clove::HorizontalStack, 460
 - clove::HtmlView, 482
 - clove::IconView, 505
 - clove::ImageView, 526
 - clove::Label, 547
 - clove::ListView, 571
 - clove::MainView, 602
 - clove::MediaPlayer, 628
 - clove::Menubar, 654
 - clove::ModalPanel, 674
 - clove::MultilineEditBox, 696
 - clove::NumericEditBox, 738
 - clove::PasswordEditBox, 764
 - clove::PopupMenu, 787
 - clove::PopupMenuButton, 808
 - clove::ProgressBar, 830
 - clove::RadioButton, 859
 - clove::RawResizeSplitter, 883
 - clove::ResizeSplitter, 904
 - clove::RichEdit, 928
 - clove::RichEditBox, 948
 - clove::ScrollView, 971
 - clove::Slider, 993
 - clove::Spacer, 1024
 - clove::TabView, 1080
 - clove::TableView, 1049
 - clove::TimeBox, 1104
 - clove::ToolBar, 1127
 - clove::TreeView, 1150
 - clove::VerticalStack, 1191
 - clove::VideoPlayer, 1215
 - clove::Widget, 1243
 - clove::Wrap, 1262

- body
 - clove::Border, [93](#)
 - clove::Dialog, [257](#)
 - clove::Expander, [354](#)
 - clove::ResizeSplitter, [904](#)
 - clove::ScrollView, [971](#)
- bodyLeft
 - clove::MainView, [602](#)
- bodyLeftViewActionLabel
 - clove::MainView, [602](#)
- bodyRight
 - clove::MainView, [602](#)
- bodyRightViewActionLabel
 - clove::MainView, [602](#)
- bodySize
 - clove::ScrollView, [971](#)
- bodyWidget
 - clove::ScrollView, [971](#)
- bodyWidgets
 - clove::ResizeSplitter, [904](#)
- browser
 - clove, [181](#)
- build
 - clove, [181](#)
- busy
 - clove::AbstractMenu, [28](#)
 - clove::AudioPlayer, [68](#)
 - clove::Border, [94](#)
 - clove::Button, [114](#)
 - clove::Carousel, [136](#)
 - clove::CheckBox, [161](#)
 - clove::DataView, [204](#)
 - clove::DateBox, [234](#)
 - clove::Dialog, [258](#)
 - clove::DropDownBox, [280](#)
 - clove::EditBox, [304](#)
 - clove::EditComboBox, [327](#)
 - clove::Expander, [354](#)
 - clove::FlatView, [390](#)
 - clove::Form, [413](#)
 - clove::Grid, [433](#)
 - clove::HorizontalStack, [460](#)
 - clove::HtmlView, [483](#)
 - clove::IconView, [506](#)
 - clove::ImageView, [526](#)
 - clove::Label, [547](#)
 - clove::ListView, [571](#)
 - clove::MainView, [603](#)
 - clove::MediaPlayer, [629](#)
 - clove::Menubar, [654](#)
 - clove::ModalPanel, [674](#)
 - clove::MultilineEditBox, [696](#)
 - clove::NumericEditBox, [739](#)
 - clove::PasswordEditBox, [764](#)
 - clove::PopupMenu, [787](#)
 - clove::PopupMenuButton, [808](#)
 - clove::ProgressBar, [830](#)
 - clove::RadioButton, [859](#)
 - clove::RawResizeSplitter, [884](#)
 - clove::ResizeSplitter, [904](#)
 - clove::RichEdit, [928](#)
 - clove::RichEditBox, [948](#)
 - clove::ScrollView, [972](#)
 - clove::Slider, [994](#)
 - clove::Spacer, [1025](#)
 - clove::TabView, [1080](#)
 - clove::TableView, [1050](#)
 - clove::TimeBox, [1104](#)
 - clove::ToolBar, [1127](#)
 - clove::TreeView, [1151](#)
 - clove::VerticalStack, [1192](#)
 - clove::VideoPlayer, [1216](#)
 - clove::Widget, [1243](#)
 - clove::Wrap, [1263](#)
- bySymbol
 - clove::Icon, [501](#)
- byUrl
 - clove::Icon, [501](#)
- canPlayType
 - clove::AudioPlayer, [68](#)
 - clove::VideoPlayer, [1216](#)
- cellGenerator
 - clove::DataView, [204](#)
 - clove::ListView, [572](#)
 - clove::TableView, [1050](#)
 - clove::TreeView, [1151](#)
- changeValue
 - clove::AjaxAsyncDatasource, [48](#)
 - clove::AsyncDatasource, [57](#)
 - clove::Datasource, [190](#)
 - clove::FilterProxyDatasource, [374](#)
 - clove::NativeDatasource, [718](#)
 - clove::ProxyDatasource, [849](#)
 - clove::SortProxyDatasource, [1013](#)
- checkable
 - clove::Button, [114](#)
 - clove::PopupMenuButton, [808](#)
- checked
 - clove::Button, [114](#)
 - clove::CheckBox, [161](#)
 - clove::PopupMenuButton, [808](#)
 - clove::RadioButton, [859](#)
- checkedCells
 - clove::DataView, [204](#)
 - clove::ListView, [572](#)
 - clove::TableView, [1050](#)
 - clove::TreeView, [1151](#)
- children
 - clove::FlatLayout, [382](#)
 - clove::FlatView, [390](#)
 - clove::Grid, [433](#)
 - clove::GridLayout, [451](#)
 - clove::HorizontalStack, [460](#)
 - clove::Layout, [565](#)
 - clove::StackLayout, [1041](#)
 - clove::VerticalStack, [1192](#)

- clove::Wrap, 1263
 - clove::WrapLayout, 1279
- childrenWidgets
 - clove::AbstractMenu, 28
 - clove::AudioPlayer, 69
 - clove::Border, 94
 - clove::Button, 114
 - clove::Carousel, 136
 - clove::CheckBox, 161
 - clove::DataView, 204
 - clove::DateBox, 234
 - clove::Dialog, 258
 - clove::DropDownBox, 280
 - clove::EditBox, 304
 - clove::EditComboBox, 327
 - clove::Expander, 354
 - clove::FlatView, 390
 - clove::Form, 413
 - clove::Grid, 433
 - clove::HorizontalStack, 460
 - clove::HtmlView, 483
 - clove::IconView, 506
 - clove::ImageView, 526
 - clove::Label, 547
 - clove::ListView, 572
 - clove::MainView, 603
 - clove::MediaPlayer, 629
 - clove::Menubar, 654
 - clove::ModalPanel, 674
 - clove::MultilineEditBox, 696
 - clove::NumericEditBox, 739
 - clove::PasswordEditBox, 764
 - clove::PopupMenu, 787
 - clove::PopupMenuButton, 809
 - clove::ProgressBar, 831
 - clove::RadioButton, 859
 - clove::RawResizeSplitter, 884
 - clove::ResizeSplitter, 905
 - clove::RichEdit, 928
 - clove::RichEditBox, 948
 - clove::ScrollView, 972
 - clove::Slider, 994
 - clove::Spacer, 1025
 - clove::TabView, 1080
 - clove::TableView, 1050
 - clove::TimeBox, 1104
 - clove::Toolbar, 1127
 - clove::TreeView, 1151
 - clove::VerticalStack, 1192
 - clove::VideoPlayer, 1216
 - clove::Widget, 1243
 - clove::Wrap, 1263
- clearCache
 - clove::AjaxAsyncDatasource, 48
 - clove::AsyncDatasource, 58
- clearChilds
 - clove::FlatLayout, 382
 - clove::FlatView, 390
 - clove::Grid, 433
 - clove::GridLayout, 451
 - clove::HorizontalStack, 461
 - clove::Layout, 565
 - clove::StackLayout, 1041
 - clove::VerticalStack, 1192
 - clove::Wrap, 1263
 - clove::WrapLayout, 1280
- close
 - clove::Dialog, 258
- clove, 177
 - browser, 181
 - build, 181
 - conversationDialog, 182
 - databind, 182
 - getByName, 183
 - i18n, 183
 - inputDialog, 183
 - Invisible, 184
 - InvisibleCollapsed, 184
 - MenuSeparator, 184
 - messageDialog, 184
 - notifications, 185
 - populateUI, 185
 - rootNameScope, 186
 - selectionDialog, 186
 - Visible, 186
- clove::AbstractMenu, 23
 - actions, 27
 - addStyleClass, 27
 - bindProperty, 28
 - busy, 28
 - childrenWidgets, 28
 - computeMinimalHeightForWidth, 28
 - computeMinimalWidth, 29
 - computePreferredHeightForWidth, 29
 - computePreferredWidth, 29
 - containingNameScope, 29
 - declareProperty, 29
 - doStandaloneResizing, 30
 - doinit, 30
 - doinitEarly, 30
 - doresize, 30
 - effectiveVisibility, 31
 - effectivelyEnabled, 31
 - enabled, 31
 - focus, 31
 - getMinimalHeightForWidth, 31
 - getMinimalWidth, 32
 - getPreferredHeightForWidth, 32
 - getPreferredWidth, 32
 - getProperty, 32
 - hasFocus, 33
 - horizontalStretchAffinity, 33
 - hstretch, 33
 - init, 33
 - innerSize, 33
 - isAlive, 34

- isStyleClass, [34](#)
- mayFocus, [34](#)
- name, [34](#)
- nameScope, [34](#)
- OnActionTriggered, [35](#)
- OnBeforeSubactionsExpanded, [35](#)
- OnDestroyed, [35](#)
- OnKeyDown, [35](#)
- OnKeyPress, [35](#)
- OnKeyUp, [35](#)
- OnPropertyChanged, [36](#)
- OnResized, [36](#)
- OnVisibilityChanged, [36](#)
- outerSize, [36](#)
- parentWidget, [36](#)
- registerBusy, [36](#)
- relayout, [37](#)
- remove, [37](#)
- removeStyleClass, [37](#)
- resize, [38](#)
- setActions, [38](#)
- setBusy, [38](#)
- setDoStandaloneResizing, [38](#)
- setEnabled, [39](#)
- setHorizontalStretchAffinity, [39](#)
- setHstretch, [39](#)
- setMayFocus, [39](#)
- setName, [40](#)
- setProperty, [40](#)
- setStrictHorizontalSizing, [40](#)
- setStrictVerticalSizing, [41](#)
- setStyle, [41](#)
- setStyleClass, [41](#)
- setStyleClassAssigned, [41](#)
- setVerticalStretchAffinity, [42](#)
- setVisibility, [42](#)
- setVstretch, [42](#)
- strictHorizontalSizing, [42](#)
- strictVerticalSizing, [43](#)
- style, [43](#)
- styleClass, [43](#)
- unregisterBusy, [43](#)
- verticalStretchAffinity, [43](#)
- visibility, [44](#)
- vstretch, [44](#)
- clove::AjaxAsyncDatasource, [44](#)
 - AjaxAsyncDatasource, [46](#)
 - async_pull, [47](#)
 - async_push, [47](#)
 - changeValue, [48](#)
 - clearCache, [48](#)
 - columnCount, [48](#)
 - columnHeadersVisible, [48](#)
 - do_async_pull, [49](#)
 - do_async_push, [49](#)
 - getAjaxId, [49](#)
 - getColumnHeader, [50](#)
 - getMetadata, [50](#)
 - getRowHeader, [50](#)
 - getValue, [51](#)
 - OnDataArrived, [51](#)
 - OnDataInsert, [51](#)
 - OnDataRemove, [51](#)
 - OnDataUpdate, [51](#)
 - OnHeaderDataInsert, [51](#), [52](#)
 - OnHeaderDataRemove, [52](#)
 - OnHeaderDataUpdate, [52](#)
 - OnHeaderVisibilityUpdated, [52](#), [53](#)
 - parent, [53](#)
 - rowCount, [53](#)
 - rowHeadersVisible, [53](#)
 - valuePointer, [54](#)
 - valuePointerNavigateInDepth, [54](#)
- clove::AsyncDatasource, [54](#)
 - async_pull, [56](#)
 - async_push, [57](#)
 - AsyncDatasource, [56](#)
 - changeValue, [57](#)
 - clearCache, [58](#)
 - columnCount, [58](#)
 - columnHeadersVisible, [58](#)
 - do_async_pull, [58](#)
 - do_async_push, [59](#)
 - getColumnHeader, [59](#)
 - getMetadata, [59](#)
 - getRowHeader, [60](#)
 - getValue, [60](#)
 - OnDataInsert, [60](#)
 - OnDataRemove, [60](#)
 - OnDataUpdate, [60](#)
 - OnHeaderDataInsert, [61](#)
 - OnHeaderDataRemove, [61](#)
 - OnHeaderDataUpdate, [61](#)
 - OnHeaderVisibilityUpdated, [61](#)
 - parent, [61](#)
 - rowCount, [62](#)
 - rowHeadersVisible, [62](#)
 - valuePointer, [62](#)
 - valuePointerNavigateInDepth, [63](#)
- clove::AudioPlayer, [63](#)
 - addStyleClass, [67](#)
 - autoPlay, [68](#)
 - bindProperty, [68](#)
 - busy, [68](#)
 - canPlayType, [68](#)
 - childrenWidgets, [69](#)
 - computeMinimalHeightForWidth, [69](#)
 - computeMinimalWidth, [69](#)
 - computePreferredHeightForWidth, [69](#)
 - computePreferredWidth, [69](#)
 - containingNameScope, [70](#)
 - currentMediaPosition, [70](#)
 - declareProperty, [70](#)
 - doStandaloneResizing, [71](#)
 - doinit, [70](#)
 - doinitEarly, [71](#)

- doresize, 71
- duration, 71
- effectiveVisibility, 71
- effectivelyEnabled, 71
- enabled, 72
- focus, 72
- getMinimalHeightForWidth, 72
- getMinimalWidth, 72
- getPreferredHeightForWidth, 72
- getPreferredWidth, 73
- getProperty, 73
- hasEnded, 73
- hasFocus, 73
- horizontalStretchAffinity, 74
- hstretch, 74
- init, 74
- innerSize, 74
- isAlive, 74
- isPaused, 75
- isStyleClass, 75
- loop, 75
- mayFocus, 75
- muted, 75
- name, 76
- nameScope, 76
- nativeNode, 76
- OnDestroyed, 76
- OnDurationChanged, 76
- OnHasEnded, 76
- OnIsPaused, 77
- OnIsPlaying, 77
- OnKeyDown, 77
- OnKeyPress, 77
- OnKeyUp, 77
- OnLoadingAborted, 78
- OnPropertyChanged, 78
- OnReadyForPlayback, 78
- OnResized, 78
- OnVisibilityChanged, 78
- outerSize, 78
- parentWidget, 79
- pause, 79
- play, 79
- preload, 79
- registerBusy, 79
- relayout, 79
- remove, 80
- removeStyleClass, 80
- resize, 80
- setAutoPlay, 80
- setBusy, 81
- setCurrentMediaPosition, 81
- setDoStandaloneResizing, 81
- setEnabled, 82
- setHorizontalStretchAffinity, 82
- setHstretch, 82
- setLoop, 82
- setMayFocus, 83
- setMuted, 83
- setName, 83
- setPreload, 84
- setProperty, 84
- setShowControls, 84
- setSource, 85
- setStrictHorizontalSizing, 85
- setStrictVerticalSizing, 85
- setStyle, 85
- setStyleClass, 86
- setStyleClassAssigned, 86
- setVerticalStretchAffinity, 86
- setVisibility, 86
- setVolume, 87
- setVstretch, 87
- showControls, 87
- source, 87
- strictHorizontalSizing, 88
- strictVerticalSizing, 88
- style, 88
- styleClass, 88
- unregisterBusy, 88
- verticalStretchAffinity, 89
- visibility, 89
- volume, 89
- vstretch, 89
- clove::Border, 90
 - addStyleClass, 93
 - bindProperty, 93
 - body, 93
 - busy, 94
 - childrenWidgets, 94
 - computeMinimalHeightForWidth, 94
 - computeMinimalWidth, 94
 - computePreferredHeightForWidth, 94
 - computePreferredWidth, 95
 - containingNameScope, 95
 - declareProperty, 95
 - doStandaloneResizing, 96
 - doinit, 95
 - doinitEarly, 96
 - doresize, 96
 - effectiveVisibility, 96
 - effectivelyEnabled, 96
 - enabled, 97
 - focus, 97
 - getMinimalHeightForWidth, 97
 - getMinimalWidth, 97
 - getPreferredHeightForWidth, 97
 - getPreferredWidth, 98
 - getProperty, 98
 - hasFocus, 98
 - horizontalStretchAffinity, 98
 - hstretch, 99
 - init, 99
 - innerSize, 99
 - isAlive, 99
 - isStyleClass, 99

- mayFocus, [100](#)
- name, [100](#)
- nameScope, [100](#)
- OnDestroyed, [100](#)
- OnKeyDown, [100](#)
- OnKeyPress, [101](#)
- OnKeyUp, [101](#)
- OnPropertyChanged, [101](#)
- OnResized, [101](#)
- OnVisibilityChanged, [101](#)
- outerSize, [101](#)
- parentWidget, [102](#)
- registerBusy, [102](#)
- relayout, [102](#)
- remove, [102](#)
- removeStyleClass, [103](#)
- resize, [103](#)
- setBody, [103](#)
- setBusy, [103](#)
- setDoStandaloneResizing, [104](#)
- setEnabled, [104](#)
- setHorizontalStretchAffinity, [104](#)
- setHstretch, [104](#)
- setMayFocus, [105](#)
- setName, [105](#)
- setProperty, [105](#)
- setStrictHorizontalSizing, [106](#)
- setStrictVerticalSizing, [106](#)
- setStyle, [106](#)
- setStyleClass, [106](#)
- setStyleClassAssigned, [107](#)
- setVerticalStretchAffinity, [107](#)
- setVisibility, [107](#)
- setVstretch, [108](#)
- strictHorizontalSizing, [108](#)
- strictVerticalSizing, [108](#)
- style, [108](#)
- styleClass, [108](#)
- unregisterBusy, [109](#)
- verticalStretchAffinity, [109](#)
- visibility, [109](#)
- vstretch, [109](#)
- clove::Button, [110](#)
 - addStyleClass, [113](#)
 - bindProperty, [113](#)
 - busy, [114](#)
 - checkable, [114](#)
 - checked, [114](#)
 - childrenWidgets, [114](#)
 - computeMinimalHeightForWidth, [114](#)
 - computeMinimalWidth, [115](#)
 - computePreferredHeightForWidth, [115](#)
 - computePreferredWidth, [115](#)
 - containingNameScope, [115](#)
 - declareProperty, [115](#)
 - doStandaloneResizing, [116](#)
 - doinit, [116](#)
 - doinitEarly, [116](#)
 - doresize, [116](#)
 - effectiveVisibility, [117](#)
 - effectivelyEnabled, [117](#)
 - enabled, [117](#)
 - focus, [117](#)
 - getMinimalHeightForWidth, [117](#)
 - getMinimalWidth, [118](#)
 - getPreferredHeightForWidth, [118](#)
 - getPreferredWidth, [118](#)
 - getProperty, [118](#)
 - hasFocus, [119](#)
 - horizontalStretchAffinity, [119](#)
 - hstretch, [119](#)
 - icon, [119](#)
 - init, [119](#)
 - innerSize, [120](#)
 - isAlive, [120](#)
 - isStyleClass, [120](#)
 - label, [120](#)
 - mayFocus, [120](#)
 - name, [120](#)
 - nameScope, [121](#)
 - OnClicked, [121](#)
 - OnDestroyed, [121](#)
 - OnKeyDown, [121](#)
 - OnKeyPress, [121](#)
 - OnKeyUp, [121](#)
 - OnPropertyChanged, [122](#)
 - OnResized, [122](#)
 - OnVisibilityChanged, [122](#)
 - outerSize, [122](#)
 - parentWidget, [122](#)
 - registerBusy, [122](#)
 - relayout, [123](#)
 - remove, [123](#)
 - removeStyleClass, [123](#)
 - resize, [124](#)
 - setBusy, [124](#)
 - setCheckable, [124](#)
 - setChecked, [124](#)
 - setDoStandaloneResizing, [125](#)
 - setEnabled, [125](#)
 - setHorizontalStretchAffinity, [125](#)
 - setHstretch, [125](#)
 - setIcon, [126](#)
 - setLabel, [126](#)
 - setMayFocus, [126](#)
 - setName, [126](#)
 - setProperty, [127](#)
 - setStrictHorizontalSizing, [127](#)
 - setStrictVerticalSizing, [127](#)
 - setStyle, [128](#)
 - setStyleClass, [128](#)
 - setStyleClassAssigned, [128](#)
 - setVerticalStretchAffinity, [128](#)
 - setVisibility, [129](#)
 - setVstretch, [129](#)
 - strictHorizontalSizing, [129](#)

- strictVerticalSizing, 129
- style, 130
- styleClass, 130
- unregisterBusy, 130
- verticalStretchAffinity, 130
- visibility, 130
- vstretch, 131
- clove::Carousel, 131
 - addStyleClass, 135
 - addTab, 135
 - bindProperty, 136
 - busy, 136
 - childrenWidgets, 136
 - computeMinimalHeightForWidth, 136
 - computeMinimalWidth, 137
 - computePreferredHeightForWidth, 137
 - computePreferredWidth, 137
 - containingNameScope, 137
 - currentTab, 138
 - declareProperty, 138
 - doStandaloneResizing, 139
 - doinit, 138
 - doinitEarly, 138
 - doresize, 139
 - effectiveVisibility, 139
 - effectivelyEnabled, 139
 - enabled, 139
 - focus, 140
 - getMinimalHeightForWidth, 140
 - getMinimalWidth, 140
 - getPreferredHeightForWidth, 140
 - getPreferredWidth, 141
 - getProperty, 141
 - hasFocus, 141
 - horizontalStretchAffinity, 141
 - hstretch, 141
 - init, 142
 - innerSize, 142
 - interval, 142
 - isAlive, 142
 - isPlaying, 142
 - isStyleClass, 143
 - mayFocus, 143
 - name, 143
 - nameScope, 143
 - next, 143
 - OnDestroyed, 144
 - OnKeyDown, 144
 - OnKeyPress, 144
 - OnKeyUp, 144
 - OnPropertyChanged, 144
 - OnResized, 144
 - OnTabCreationRequested, 145
 - OnVisibilityChanged, 145
 - outerSize, 145
 - parentWidget, 145
 - pause, 145
 - play, 145
 - registerBusy, 146
 - layout, 146
 - remove, 146
 - removeStyleClass, 146
 - resize, 147
 - setBusy, 147
 - setCurrentTab, 147
 - setDoStandaloneResizing, 147
 - setEnabled, 148
 - setHorizontalStretchAffinity, 148
 - setHstretch, 148
 - setInterval, 148
 - setMayFocus, 149
 - setName, 149
 - setProperty, 149
 - setStrictHorizontalSizing, 150
 - setStrictVerticalSizing, 150
 - setStyle, 150
 - setStyleClass, 150
 - setStyleClassAssigned, 151
 - setSwitchInvisibleAnimationDuration, 151
 - setSwitchInvisibleAnimationName, 151
 - setSwitchVisibleAnimationDuration, 152
 - setSwitchVisibleAnimationName, 152
 - setTabBarLocation, 152
 - setTabs, 152
 - setUserMayAddTabs, 153
 - setVerticalStretchAffinity, 153
 - setVisibility, 153
 - setVstretch, 154
 - strictHorizontalSizing, 154
 - strictVerticalSizing, 154
 - style, 154
 - styleClass, 154
 - switchInvisibleAnimationDuration, 155
 - switchInvisibleAnimationName, 155
 - switchVisibleAnimationDuration, 155
 - switchVisibleAnimationName, 155
 - tabBarLocation, 155
 - tabs, 155
 - unregisterBusy, 155
 - userMayAddTabs, 156
 - verticalStretchAffinity, 156
 - visibility, 156
 - vstretch, 156
- clove::CheckButton, 157
 - addStyleClass, 160
 - bindProperty, 160
 - busy, 161
 - checked, 161
 - childrenWidgets, 161
 - computeMinimalHeightForWidth, 161
 - computeMinimalWidth, 161
 - computePreferredHeightForWidth, 162
 - computePreferredWidth, 162
 - containingNameScope, 162
 - declareProperty, 162
 - doStandaloneResizing, 163

- doinit, [163](#)
- doinitEarly, [163](#)
- doresize, [163](#)
- effectiveVisibility, [164](#)
- effectivelyEnabled, [163](#)
- enabled, [164](#)
- focus, [164](#)
- getMinimalHeightForWidth, [164](#)
- getMinimalWidth, [164](#)
- getPreferredHeightForWidth, [165](#)
- getPreferredWidth, [165](#)
- getProperty, [165](#)
- hasFocus, [166](#)
- horizontalStretchAffinity, [166](#)
- hstretch, [166](#)
- init, [166](#)
- innerSize, [166](#)
- isAlive, [166](#)
- isStyleClass, [167](#)
- label, [167](#)
- mayFocus, [167](#)
- name, [167](#)
- nameScope, [167](#)
- OnChanged, [168](#)
- OnClicked, [168](#)
- OnDestroyed, [168](#)
- OnKeyDown, [168](#)
- OnKeyPress, [168](#)
- OnKeyUp, [168](#)
- OnPropertyChanged, [169](#)
- OnResized, [169](#)
- OnVisibilityChanged, [169](#)
- outerSize, [169](#)
- parentWidget, [169](#)
- registerBusy, [169](#)
- relayout, [170](#)
- remove, [170](#)
- removeStyleClass, [170](#)
- resize, [171](#)
- setBusy, [171](#)
- setChecked, [171](#)
- setDoStandaloneResizing, [171](#)
- setEnabled, [172](#)
- setHorizontalStretchAffinity, [172](#)
- setHstretch, [172](#)
- setLabel, [172](#)
- setMayFocus, [173](#)
- setName, [173](#)
- setProperty, [173](#)
- setStrictHorizontalSizing, [174](#)
- setStrictVerticalSizing, [174](#)
- setStyle, [174](#)
- setStyleClass, [174](#)
- setStyleClassAssigned, [175](#)
- setVerticalStretchAffinity, [175](#)
- setVisibility, [175](#)
- setVstretch, [175](#)
- strictHorizontalSizing, [176](#)
- strictVerticalSizing, [176](#)
- style, [176](#)
- styleClass, [176](#)
- unregisterBusy, [176](#)
- verticalStretchAffinity, [177](#)
- visibility, [177](#)
- vstretch, [177](#)
- clove::DataBinding, [187](#)
 - bind, [187](#)
 - DataBinding, [187](#)
 - unbind, [188](#)
- clove::DataBindingConverter, [188](#)
 - convertFrom, [188](#)
 - convertTo, [189](#)
- clove::DataView, [197](#)
 - addStyleClass, [203](#)
 - allowChecking, [203](#)
 - allowSelection, [203](#)
 - alwaysAllocateExpanderSpace, [203](#)
 - bindProperty, [203](#)
 - busy, [204](#)
 - cellGenerator, [204](#)
 - checkedCells, [204](#)
 - childrenWidgets, [204](#)
 - collapseCell, [204](#)
 - columnsResizable, [205](#)
 - computeMinimalHeightForWidth, [205](#)
 - computeMinimalWidth, [205](#)
 - computePreferredHeightForWidth, [205](#)
 - computePreferredWidth, [206](#)
 - containingNameScope, [206](#)
 - dataViewProcessor, [206](#)
 - datasource, [206](#)
 - declareProperty, [206](#)
 - doStandaloneResizing, [207](#)
 - doinit, [207](#)
 - doinitEarly, [207](#)
 - doresize, [207](#)
 - editCell, [207](#)
 - editOnGesture, [208](#)
 - effectiveVisibility, [208](#)
 - effectivelyEnabled, [208](#)
 - enabled, [208](#)
 - expandCell, [208](#)
 - expandCellRecursive, [209](#)
 - focus, [209](#)
 - getMinimalHeightForWidth, [209](#)
 - getMinimalWidth, [209](#)
 - getPreferredHeightForWidth, [209](#)
 - getPreferredWidth, [210](#)
 - getProperty, [210](#)
 - granularity, [210](#)
 - gridVisible, [210](#)
 - hasFocus, [211](#)
 - headersource, [211](#)
 - hideExpanders, [211](#)
 - horizontalStretchAffinity, [211](#)
 - hstretch, [211](#)

- init, [211](#)
- innerSize, [212](#)
- isAlive, [212](#)
- isCellChecked, [212](#)
- isCellExpanded, [212](#)
- isCellSelected, [213](#)
- isStyleClass, [213](#)
- mayFocus, [213](#)
- name, [213](#)
- nameScope, [213](#)
- nodeActivationNeedsDoubleClick, [213](#)
- OnDestroyed, [214](#)
- OnKeyDown, [214](#)
- OnKeyPress, [214](#)
- OnKeyUp, [214](#)
- OnPropertyChanged, [214](#)
- OnResized, [214](#)
- OnSelectionChanged, [215](#)
- OnVisibilityChanged, [215](#)
- outerSize, [215](#)
- parentWidget, [215](#)
- registerBusy, [215](#)
- relayout, [215](#)
- remove, [216](#)
- removeStyleClass, [216](#)
- resize, [216](#)
- rowsResizable, [216](#)
- selectCell, [217](#)
- selection, [217](#)
- setAllowChecking, [217](#)
- setAllowSelection, [217](#)
- setAlwaysAllocateExpanderSpace, [218](#)
- setBusy, [218](#)
- setCellChecked, [218](#)
- setCellGenerator, [218](#)
- setCheckedCells, [219](#)
- setColumnsResizable, [219](#)
- setDataViewProcessor, [220](#)
- setDatasource, [219](#)
- setDoStandaloneResizing, [220](#)
- setEditOnGesture, [220](#)
- setEnabled, [220](#)
- setGranularity, [221](#)
- setGridVisible, [221](#)
- setHeadersource, [221](#)
- setHideExpanders, [222](#)
- setHorizontalStretchAffinity, [222](#)
- setHstretch, [222](#)
- setMayFocus, [222](#)
- setName, [223](#)
- setNodeActivationNeedsDoubleClick, [223](#)
- setProperty, [223](#)
- setRowsResizable, [224](#)
- setShowChangeMenu, [224](#)
- setShowOnlyFirstColumn, [224](#)
- setStrictHorizontalSizing, [224](#)
- setStrictVerticalSizing, [225](#)
- setStyle, [225](#)
- setStyleClass, [225](#)
- setStyleClassAssigned, [226](#)
- setVerticalStretchAffinity, [226](#)
- setVisibility, [226](#)
- setVstretch, [226](#)
- showChangeMenu, [227](#)
- showOnlyFirstColumn, [227](#)
- strictHorizontalSizing, [227](#)
- strictVerticalSizing, [227](#)
- style, [228](#)
- styleClass, [228](#)
- unregisterBusy, [228](#)
- verticalStretchAffinity, [228](#)
- visibility, [228](#)
- vstretch, [229](#)
- clove::Datasource, [189](#)
 - changeValue, [190](#)
 - columnCount, [190](#)
 - getMetadata, [191](#)
 - getValue, [191](#)
 - OnDataInsert, [191](#)
 - OnDataRemove, [191](#)
 - OnDataUpdate, [191](#)
 - parent, [192](#)
 - rowCount, [192](#)
 - valuePointer, [192](#)
 - valuePointerNavigateInDepth, [193](#)
- clove::DatasourceValuePointer, [193](#)
 - backendObject, [194](#)
 - columnCount, [194](#)
 - datasource, [195](#)
 - DatasourceValuePointer, [194](#)
 - equals, [195](#)
 - fromPath, [195](#)
 - getValue, [195](#)
 - icol, [196](#)
 - irow, [196](#)
 - parent, [196](#)
 - rowCount, [196](#)
 - sibling, [196](#)
 - toPath, [197](#)
 - toString, [197](#)
- clove::DateBox, [229](#)
 - addStyleClass, [233](#)
 - autocompletionFilter, [233](#)
 - autocompletionItems, [233](#)
 - autocompletionOpenForNoText, [234](#)
 - bindProperty, [234](#)
 - busy, [234](#)
 - childrenWidgets, [234](#)
 - computeMinimalHeightForWidth, [234](#)
 - computeMinimalWidth, [236](#)
 - computePreferredHeightForWidth, [236](#)
 - computePreferredWidth, [236](#)
 - containingNameScope, [236](#)
 - date, [236](#)
 - declareProperty, [237](#)
 - doStandaloneResizing, [238](#)

doinit, [237](#)
doinitEarly, [237](#)
doresize, [237](#)
effectiveVisibility, [238](#)
effectivelyEnabled, [238](#)
enabled, [238](#)
focus, [238](#)
getMinimalHeightForWidth, [238](#)
getMinimalWidth, [239](#)
getPreferredHeightForWidth, [239](#)
getPreferredWidth, [239](#)
getProperty, [239](#)
hasFocus, [240](#)
hintText, [240](#)
horizontalStretchAffinity, [240](#)
hstretch, [240](#)
init, [240](#)
innerSize, [241](#)
isAlive, [241](#)
isStyleClass, [241](#)
mayFocus, [241](#)
name, [241](#)
nameScope, [241](#)
OnChanged, [242](#)
OnDestroyed, [242](#)
OnKeyDown, [242](#)
OnKeyPress, [242](#)
OnKeyUp, [242](#)
OnPopupTextSelected, [242](#)
OnPropertyChanged, [243](#)
OnResized, [243](#)
OnVisibilityChanged, [243](#)
outerSize, [243](#)
parentWidget, [243](#)
readOnly, [243](#)
registerBusy, [244](#)
relayout, [244](#)
remove, [244](#)
removeStyleClass, [244](#)
resize, [245](#)
setAutocompletionFilter, [245](#)
setAutocompletionItems, [245](#)
setAutocompletionOpenForNoText, [245](#)
setBusy, [246](#)
setDate, [246](#)
setDoStandaloneResizing, [246](#)
setEnabled, [246](#)
setHintText, [247](#)
setHorizontalStretchAffinity, [247](#)
setHstretch, [247](#)
setMayFocus, [248](#)
setName, [248](#)
setProperty, [248](#)
setReadOnly, [249](#)
setStrictHorizontalSizing, [249](#)
setStrictVerticalSizing, [249](#)
setStyle, [249](#)
setStyleClass, [250](#)
setStyleClassAssigned, [250](#)
setText, [250](#)
setTextSelection, [250](#)
setVerticalStretchAffinity, [251](#)
setVisibility, [251](#)
setVstretch, [251](#)
strictHorizontalSizing, [252](#)
strictVerticalSizing, [252](#)
style, [252](#)
styleClass, [252](#)
text, [252](#)
textSelection, [252](#)
unregisterBusy, [252](#)
verticalStretchAffinity, [253](#)
visibility, [253](#)
vstretch, [253](#)
clove::Dialog, [253](#)
 addStyleClass, [257](#)
 bindProperty, [257](#)
 body, [257](#)
 busy, [258](#)
 childrenWidgets, [258](#)
 close, [258](#)
 computeMinimalHeightForWidth, [258](#)
 computeMinimalWidth, [258](#)
 computePreferredHeightForWidth, [259](#)
 computePreferredWidth, [259](#)
 containingNameScope, [259](#)
 declareProperty, [259](#)
 doStandaloneResizing, [260](#)
 doinit, [260](#)
 doinitEarly, [260](#)
 doresize, [260](#)
 effectiveVisibility, [261](#)
 effectivelyEnabled, [260](#)
 enabled, [261](#)
 focus, [261](#)
 getMinimalHeightForWidth, [261](#)
 getMinimalWidth, [261](#)
 getPreferredHeightForWidth, [262](#)
 getPreferredWidth, [262](#)
 getProperty, [262](#)
 hasFocus, [263](#)
 horizontalStretchAffinity, [263](#)
 hstretch, [263](#)
 init, [263](#)
 innerSize, [263](#)
 isAlive, [263](#)
 isStyleClass, [264](#)
 mayFocus, [264](#)
 name, [264](#)
 nameScope, [264](#)
 OnDestroyed, [264](#)
 OnKeyDown, [265](#)
 OnKeyPress, [265](#)
 OnKeyUp, [265](#)
 OnPropertyChanged, [265](#)
 OnResized, [265](#)

- OnVisibilityChanged, 265
- outerSize, 266
- parentWidget, 266
- registerBusy, 266
- relayout, 266
- remove, 266
- removeStyleClass, 267
- resize, 267
- setBody, 267
- setBusy, 267
- setDoStandaloneResizing, 268
- setEnabled, 268
- setHorizontalStretchAffinity, 268
- setHstretch, 268
- setMayFocus, 269
- setName, 269
- setProperty, 269
- setStrictHorizontalSizing, 270
- setStrictVerticalSizing, 270
- setStyle, 270
- setStyleClass, 270
- setStyleClassAssigned, 271
- setTitle, 271
- setTitleicon, 271
- setVerticalStretchAffinity, 272
- setVisibility, 272
- setVstretch, 272
- show, 272
- strictHorizontalSizing, 273
- strictVerticalSizing, 273
- style, 273
- styleClass, 273
- title, 273
- titleicon, 274
- unregisterBusy, 274
- verticalStretchAffinity, 274
- visibility, 274
- vstretch, 274
- clove::DropDownBox, 275
 - addStyleClass, 279
 - autocompleteFilter, 279
 - autocompleteItems, 279
 - autocompleteOpenForNoText, 279
 - bindProperty, 279
 - busy, 280
 - childrenWidgets, 280
 - computeMinimalHeightForWidth, 280
 - computeMinimalWidth, 280
 - computePreferredHeightForWidth, 281
 - computePreferredWidth, 281
 - containingNameScope, 281
 - declareProperty, 281
 - doStandaloneResizing, 282
 - doinit, 282
 - doinitEarly, 282
 - doresize, 282
 - effectiveVisibility, 283
 - effectivelyEnabled, 282
 - enabled, 283
 - focus, 283
 - getMinimalHeightForWidth, 283
 - getMinimalWidth, 283
 - getPreferredHeightForWidth, 284
 - getPreferredWidth, 284
 - getProperty, 284
 - hasFocus, 285
 - hintText, 285
 - horizontalStretchAffinity, 285
 - hstretch, 285
 - init, 285
 - innerSize, 286
 - isAlive, 286
 - isStyleClass, 286
 - mayFocus, 286
 - name, 286
 - nameScope, 286
 - OnChanged, 287
 - OnDestroyed, 287
 - OnKeyDown, 287
 - OnKeyPress, 287
 - OnKeyUp, 287
 - OnPopupTextSelected, 287
 - OnPropertyChanged, 288
 - OnResized, 288
 - OnVisibilityChanged, 288
 - outerSize, 288
 - parentWidget, 288
 - popupItems, 288
 - readOnly, 289
 - registerBusy, 289
 - relayout, 289
 - remove, 289
 - removeStyleClass, 290
 - resize, 290
 - setAutocompletionFilter, 290
 - setAutocompletionItems, 290
 - setAutocompletionOpenForNoText, 291
 - setBusy, 291
 - setDoStandaloneResizing, 291
 - setEnabled, 291
 - setHintText, 292
 - setHorizontalStretchAffinity, 292
 - setHstretch, 292
 - setMayFocus, 293
 - setName, 293
 - setPopupItems, 293
 - setProperty, 293
 - setReadOnly, 294
 - setStrictHorizontalSizing, 294
 - setStrictVerticalSizing, 294
 - setStyle, 295
 - setStyleClass, 295
 - setStyleClassAssigned, 295
 - setText, 295
 - setTextSelection, 296
 - setVerticalStretchAffinity, 296

- setVisibility, 296
- setVstretch, 297
- strictHorizontalSizing, 297
- strictVerticalSizing, 297
- style, 297
- styleClass, 297
- text, 298
- textSelection, 298
- unregisterBusy, 298
- verticalStretchAffinity, 298
- visibility, 298
- vstretch, 299
- clove::EditBox, 299
 - addStyleClass, 303
 - autocompletionFilter, 303
 - autocompletionItems, 303
 - autocompletionOpenForNoText, 303
 - bindProperty, 304
 - busy, 304
 - childrenWidgets, 304
 - computeMinimalHeightForWidth, 304
 - computeMinimalWidth, 305
 - computePreferredHeightForWidth, 305
 - computePreferredWidth, 305
 - containingNameScope, 305
 - declareProperty, 305
 - doStandaloneResizing, 306
 - doinit, 306
 - doinitEarly, 306
 - doresize, 306
 - effectiveVisibility, 307
 - effectivelyEnabled, 307
 - enabled, 307
 - focus, 307
 - getMinimalHeightForWidth, 307
 - getMinimalWidth, 308
 - getPreferredHeightForWidth, 308
 - getPreferredWidth, 308
 - getProperty, 308
 - hasFocus, 309
 - hintText, 309
 - horizontalStretchAffinity, 309
 - hstretch, 309
 - init, 309
 - innerSize, 310
 - isAlive, 310
 - isStyleClass, 310
 - mayFocus, 310
 - name, 310
 - nameScope, 310
 - OnChanged, 311
 - OnDestroyed, 311
 - OnKeyDown, 311
 - OnKeyPress, 311
 - OnKeyUp, 311
 - OnPopupTextSelected, 311
 - OnPropertyChanged, 312
 - OnResized, 312
 - OnVisibilityChanged, 312
 - outerSize, 312
 - parentWidget, 312
 - readOnly, 312
 - registerBusy, 313
 - relayout, 313
 - remove, 313
 - removeStyleClass, 313
 - resize, 314
 - setAutocompletionFilter, 314
 - setAutocompletionItems, 314
 - setAutocompletionOpenForNoText, 314
 - setBusy, 315
 - setDoStandaloneResizing, 315
 - setEnabled, 315
 - setHintText, 315
 - setHorizontalStretchAffinity, 316
 - setHstretch, 316
 - setMayFocus, 316
 - setName, 317
 - setProperty, 317
 - setReadOnly, 317
 - setStrictHorizontalSizing, 318
 - setStrictVerticalSizing, 318
 - setStyle, 318
 - setStyleClass, 318
 - setStyleClassAssigned, 319
 - setText, 319
 - setTextSelection, 319
 - setVerticalStretchAffinity, 320
 - setVisibility, 320
 - setVstretch, 320
 - strictHorizontalSizing, 320
 - strictVerticalSizing, 320
 - style, 321
 - styleClass, 321
 - text, 321
 - textSelection, 321
 - unregisterBusy, 321
 - verticalStretchAffinity, 322
 - visibility, 322
 - vstretch, 322
- clove::EditComboBox, 322
 - addStyleClass, 326
 - autocompletionFilter, 327
 - autocompletionItems, 327
 - autocompletionOpenForNoText, 327
 - bindProperty, 327
 - busy, 327
 - childrenWidgets, 327
 - computeMinimalHeightForWidth, 328
 - computeMinimalWidth, 328
 - computePreferredHeightForWidth, 328
 - computePreferredWidth, 328
 - containingNameScope, 329
 - declareProperty, 329
 - doStandaloneResizing, 330
 - doinit, 329

- doinitEarly, 329
- doresize, 329
- effectiveVisibility, 330
- effectivelyEnabled, 330
- enabled, 330
- focus, 330
- getMinimalHeightForWidth, 330
- getMinimalWidth, 331
- getPreferredHeightForWidth, 331
- getPreferredWidth, 331
- getProperty, 331
- hasFocus, 332
- hintText, 332
- horizontalStretchAffinity, 332
- hstretch, 332
- init, 332
- innerSize, 333
- isAlive, 333
- isStyleClass, 333
- mayFocus, 333
- name, 333
- nameScope, 333
- OnChanged, 334
- OnDestroyed, 334
- OnKeyDown, 334
- OnKeyPress, 334
- OnKeyUp, 334
- OnPopupTextSelected, 334
- OnPropertyChanged, 335
- OnResized, 335
- OnVisibilityChanged, 335
- outerSize, 335
- parentWidget, 335
- popupItems, 335
- readOnly, 336
- registerBusy, 336
- relayout, 336
- remove, 336
- removeStyleClass, 337
- resize, 337
- setAutocompletionFilter, 337
- setAutocompletionItems, 337
- setAutocompletionOpenForNoText, 338
- setBusy, 338
- setDoStandaloneResizing, 338
- setEnabled, 338
- setHintText, 339
- setHorizontalStretchAffinity, 339
- setHstretch, 339
- setMayFocus, 340
- setName, 340
- setPopupItems, 340
- setProperty, 340
- setReadOnly, 341
- setStrictHorizontalSizing, 341
- setStrictVerticalSizing, 341
- setStyle, 342
- setStyleClass, 342
- setStyleClassAssigned, 342
- setText, 342
- setTextSelection, 343
- setVerticalStretchAffinity, 343
- setVisibility, 343
- setVstretch, 344
- strictHorizontalSizing, 344
- strictVerticalSizing, 344
- style, 344
- styleClass, 344
- text, 345
- textSelection, 345
- unregisterBusy, 345
- verticalStretchAffinity, 345
- visibility, 345
- vstretch, 346
- clove::Event, 346
 - addHandler, 347
 - addHandlerOnce, 347
 - Event, 346
 - hasHandlers, 347
 - removeHandler, 347
 - trigger, 348
- clove::EventArgs, 348
 - EventArgs, 349
 - skipExecution, 349
- clove::Expander, 349
 - addStyleClass, 353
 - bindProperty, 353
 - body, 354
 - busy, 354
 - childrenWidgets, 354
 - collapsedIcon, 354
 - collapsedLabel, 354
 - computeMinimalHeightForWidth, 354
 - computeMinimalWidth, 355
 - computePreferredHeightForWidth, 355
 - computePreferredWidth, 355
 - containingNameScope, 355
 - declareProperty, 355
 - doStandaloneResizing, 356
 - doinit, 356
 - doinitEarly, 356
 - doresize, 356
 - effectiveVisibility, 357
 - effectivelyEnabled, 357
 - enabled, 357
 - expandedIcon, 357
 - expandedLabel, 357
 - focus, 357
 - getMinimalHeightForWidth, 357
 - getMinimalWidth, 358
 - getPreferredHeightForWidth, 358
 - getPreferredWidth, 358
 - getProperty, 358
 - hasFocus, 359
 - horizontalStretchAffinity, 359
 - hstretch, 359

- init, [359](#)
- innerSize, [359](#)
- isAlive, [360](#)
- isExpanded, [360](#)
- isStyleClass, [360](#)
- mayFocus, [360](#)
- name, [360](#)
- nameScope, [360](#)
- OnDestroyed, [361](#)
- OnKeyDown, [361](#)
- OnKeyPress, [361](#)
- OnKeyUp, [361](#)
- OnPropertyChanged, [361](#)
- OnResized, [361](#)
- OnVisibilityChanged, [362](#)
- outerSize, [362](#)
- parentWidget, [362](#)
- registerBusy, [362](#)
- relayout, [362](#)
- remove, [362](#)
- removeStyleClass, [363](#)
- resize, [363](#)
- setBody, [363](#)
- setBusy, [363](#)
- setCollapsedIcon, [364](#)
- setCollapsedLabel, [364](#)
- setDoStandaloneResizing, [364](#)
- setEnabled, [364](#)
- setExpandedIcon, [365](#)
- setExpandedLabel, [365](#)
- setHorizontalStretchAffinity, [365](#)
- setHstretch, [366](#)
- setIsExpanded, [366](#)
- setMayFocus, [366](#)
- setName, [366](#)
- setProperty, [367](#)
- setStrictHorizontalSizing, [367](#)
- setStrictVerticalSizing, [367](#)
- setStyle, [368](#)
- setStyleClass, [368](#)
- setStyleClassAssigned, [368](#)
- setVerticalStretchAffinity, [368](#)
- setVisibility, [369](#)
- setVstretch, [369](#)
- strictHorizontalSizing, [369](#)
- strictVerticalSizing, [369](#)
- style, [370](#)
- styleClass, [370](#)
- unregisterBusy, [370](#)
- verticalStretchAffinity, [370](#)
- visibility, [370](#)
- vstretch, [371](#)
- clove::FilterProxyDatasource, [371](#)
 - changeValue, [374](#)
 - columnCount, [374](#)
 - columnHeadersVisible, [374](#)
 - FilterProxyDatasource, [372](#)
 - getColumnHeader, [375](#)
 - getMetadata, [375](#)
 - getRowHeader, [375](#)
 - getValue, [376](#)
 - nativePointerToProxyPointer, [376](#)
 - OnDataInsert, [376](#)
 - OnDataRemove, [376](#)
 - OnDataUpdate, [376](#)
 - OnHeaderDataInsert, [377](#)
 - OnHeaderDataRemove, [377](#)
 - OnHeaderDataUpdate, [377](#)
 - OnHeaderVisibilityUpdated, [377](#)
 - parent, [377](#)
 - proxyPointerToNativePointer, [378](#)
 - refresh, [378](#)
 - rowCount, [378](#)
 - rowHeadersVisible, [378](#)
 - setColumnFilter, [379](#)
 - setDatasource, [379](#)
 - setRowFilter, [379](#)
 - valuePointer, [379](#)
 - valuePointerNavigateInDepth, [380](#)
- clove::FlatLayout, [380](#)
 - addChild, [381](#)
 - children, [382](#)
 - clearChilds, [382](#)
 - collapseHidden, [382](#)
 - currentView, [382](#)
 - setChildren, [382](#)
 - setCollapseHidden, [383](#)
 - setCurrentView, [383](#)
 - setSwitchInvisibleAnimationDuration, [383](#)
 - setSwitchInvisibleAnimationName, [383](#)
 - setSwitchVisibleAnimationDuration, [384](#)
 - setSwitchVisibleAnimationName, [384](#)
 - switchInvisibleAnimationDuration, [384](#)
 - switchInvisibleAnimationName, [384](#)
 - switchVisibleAnimationDuration, [385](#)
 - switchVisibleAnimationName, [385](#)
- clove::FlatView, [385](#)
 - addChild, [389](#)
 - addStyleClass, [389](#)
 - bindProperty, [390](#)
 - busy, [390](#)
 - children, [390](#)
 - childrenWidgets, [390](#)
 - clearChilds, [390](#)
 - collapseHidden, [391](#)
 - computeMinimalHeightForWidth, [391](#)
 - computeMinimalWidth, [391](#)
 - computePreferredHeightForWidth, [391](#)
 - computePreferredWidth, [392](#)
 - containingNameScope, [392](#)
 - currentView, [392](#)
 - declareProperty, [392](#)
 - doStandaloneResizing, [393](#)
 - doinit, [392](#)
 - doinitEarly, [393](#)
 - doresize, [393](#)

effectiveVisibility, 393
 effectivelyEnabled, 393
 enabled, 394
 focus, 394
 getMinimalHeightForWidth, 394
 getMinimalWidth, 394
 getPreferredHeightForWidth, 394
 getPreferredWidth, 395
 getProperty, 395
 hasFocus, 395
 horizontalStretchAffinity, 395
 hstretch, 396
 init, 396
 innerSize, 396
 isAlive, 396
 isStyleClass, 396
 mayFocus, 397
 name, 397
 nameScope, 397
 OnDestroyed, 397
 OnKeyDown, 397
 OnKeyPress, 398
 OnKeyUp, 398
 OnPropertyChanged, 398
 OnResized, 398
 OnVisibilityChanged, 398
 outerSize, 398
 parentWidget, 399
 registerBusy, 399
 relayout, 399
 remove, 399
 removeStyleClass, 400
 resize, 400
 setBusy, 400
 setChildren, 400
 setCollapseHidden, 401
 setCurrentView, 401
 setDoStandaloneResizing, 401
 setEnabled, 401
 setHorizontalStretchAffinity, 402
 setHstretch, 402
 setMayFocus, 402
 setName, 403
 setProperty, 403
 setStrictHorizontalSizing, 403
 setStrictVerticalSizing, 404
 setStyle, 404
 setStyleClass, 404
 setStyleClassAssigned, 404
 setSwitchInvisibleAnimationDuration, 405
 setSwitchInvisibleAnimationName, 405
 setSwitchVisibleAnimationDuration, 405
 setSwitchVisibleAnimationName, 405
 setVerticalStretchAffinity, 406
 setVisibility, 406
 setVstretch, 406
 strictHorizontalSizing, 407
 strictVerticalSizing, 407
 style, 407
 styleClass, 407
 switchInvisibleAnimationDuration, 407
 switchInvisibleAnimationName, 407
 switchVisibleAnimationDuration, 408
 switchVisibleAnimationName, 408
 unregisterBusy, 408
 verticalStretchAffinity, 408
 visibility, 408
 vstretch, 409
 clove::Form, 409
 addStyleClass, 412
 bindProperty, 413
 busy, 413
 childrenWidgets, 413
 computeMinimalHeightForWidth, 413
 computeMinimalWidth, 413
 computePreferredHeightForWidth, 414
 computePreferredWidth, 414
 containingNameScope, 414
 declareProperty, 414
 doStandaloneResizing, 415
 doinit, 415
 doinitEarly, 415
 doresize, 415
 effectiveVisibility, 416
 effectivelyEnabled, 415
 enabled, 416
 focus, 416
 getMinimalHeightForWidth, 416
 getMinimalWidth, 416
 getPreferredHeightForWidth, 417
 getPreferredWidth, 417
 getProperty, 417
 hasFocus, 418
 horizontalStretchAffinity, 418
 hstretch, 418
 init, 418
 innerSize, 418
 isAlive, 418
 isStyleClass, 419
 mayFocus, 419
 name, 419
 nameScope, 419
 OnDestroyed, 419
 OnKeyDown, 420
 OnKeyPress, 420
 OnKeyUp, 420
 OnPropertyChanged, 420
 OnResized, 420
 OnVisibilityChanged, 420
 outerSize, 421
 parentWidget, 421
 registerBusy, 421
 relayout, 421
 remove, 421
 removeStyleClass, 422
 resize, 422

- sections, [422](#)
- setBusy, [422](#)
- setDoStandaloneResizing, [423](#)
- setEnabled, [423](#)
- setHorizontalStretchAffinity, [423](#)
- setHstretch, [423](#)
- setMayFocus, [424](#)
- setName, [424](#)
- setProperty, [424](#)
- setSections, [425](#)
- setStrictHorizontalSizing, [425](#)
- setStrictVerticalSizing, [425](#)
- setStyle, [425](#)
- setStyleClass, [426](#)
- setStyleClassAssigned, [426](#)
- setVerticalStretchAffinity, [426](#)
- setVisibility, [426](#)
- setVstretch, [427](#)
- strictHorizontalSizing, [427](#)
- strictVerticalSizing, [427](#)
- style, [427](#)
- styleClass, [427](#)
- unregisterBusy, [428](#)
- verticalStretchAffinity, [428](#)
- visibility, [428](#)
- vstretch, [428](#)
- clove::Grid, [429](#)
 - addChild, [432](#)
 - addStyleClass, [432](#)
 - bindProperty, [433](#)
 - busy, [433](#)
 - children, [433](#)
 - childrenWidgets, [433](#)
 - clearChilds, [433](#)
 - computeMinimalHeightForWidth, [434](#)
 - computeMinimalWidth, [434](#)
 - computePreferredHeightForWidth, [434](#)
 - computePreferredWidth, [434](#)
 - containingNameScope, [435](#)
 - declareProperty, [435](#)
 - doStandaloneResizing, [436](#)
 - doinit, [435](#)
 - doinitEarly, [435](#)
 - doresize, [435](#)
 - effectiveVisibility, [436](#)
 - effectivelyEnabled, [436](#)
 - enabled, [436](#)
 - focus, [436](#)
 - getMinimalHeightForWidth, [436](#)
 - getMinimalWidth, [437](#)
 - getPreferredHeightForWidth, [437](#)
 - getPreferredWidth, [437](#)
 - getProperty, [437](#)
 - hasFocus, [438](#)
 - horizontalStretchAffinity, [438](#)
 - hstretch, [438](#)
 - init, [438](#)
 - innerSize, [438](#)
 - insertColumn, [439](#)
 - insertRow, [439](#)
 - isAlive, [439](#)
 - isStyleClass, [439](#)
 - mayFocus, [440](#)
 - name, [440](#)
 - nameScope, [440](#)
 - OnDestroyed, [440](#)
 - OnKeyDown, [440](#)
 - OnKeyPress, [440](#)
 - OnKeyUp, [441](#)
 - OnPropertyChanged, [441](#)
 - OnResized, [441](#)
 - OnVisibilityChanged, [441](#)
 - outerSize, [441](#)
 - parentWidget, [442](#)
 - registerBusy, [442](#)
 - relayout, [442](#)
 - remove, [442](#)
 - removeColumn, [443](#)
 - removeRow, [443](#)
 - removeStyleClass, [443](#)
 - resize, [443](#)
 - setBusy, [444](#)
 - setChildren, [444](#)
 - setDoStandaloneResizing, [444](#)
 - setEnabled, [444](#)
 - setHorizontalStretchAffinity, [445](#)
 - setHstretch, [445](#)
 - setMayFocus, [445](#)
 - setName, [446](#)
 - setProperty, [446](#)
 - setStrictHorizontalSizing, [446](#)
 - setStrictVerticalSizing, [447](#)
 - setStyle, [447](#)
 - setStyleClass, [447](#)
 - setStyleClassAssigned, [447](#)
 - setVerticalStretchAffinity, [448](#)
 - setVisibility, [448](#)
 - setVstretch, [448](#)
 - strictHorizontalSizing, [448](#)
 - strictVerticalSizing, [449](#)
 - style, [449](#)
 - styleClass, [449](#)
 - unregisterBusy, [449](#)
 - verticalStretchAffinity, [449](#)
 - visibility, [450](#)
 - vstretch, [450](#)
- clove::GridLayout, [450](#)
 - addChild, [451](#)
 - children, [451](#)
 - clearChilds, [451](#)
 - insertColumn, [452](#)
 - insertRow, [452](#)
 - removeColumn, [452](#)
 - removeRow, [452](#)
 - setChildren, [453](#)
- clove::Headersource, [453](#)

- columnHeadersVisible, [454](#)
- getColumnHeader, [454](#)
- getRowHeader, [455](#)
- OnHeaderDataInsert, [455](#)
- OnHeaderDataRemove, [455](#)
- OnHeaderDataUpdate, [455](#)
- OnHeaderVisibilityUpdated, [455](#)
- rowHeadersVisible, [455](#)
- clove::HorizontalStack, [456](#)
 - addChild, [459](#)
 - addStyleClass, [460](#)
 - bindProperty, [460](#)
 - busy, [460](#)
 - children, [460](#)
 - childrenWidgets, [460](#)
 - clearChilds, [461](#)
 - cols, [461](#)
 - computeMinimalHeightForWidth, [461](#)
 - computeMinimalWidth, [461](#)
 - computePreferredHeightForWidth, [461](#)
 - computePreferredWidth, [462](#)
 - containingNameScope, [462](#)
 - declareProperty, [462](#)
 - doStandaloneResizing, [463](#)
 - doinit, [462](#)
 - doinitEarly, [463](#)
 - doresize, [463](#)
 - effectiveVisibility, [463](#)
 - effectivelyEnabled, [463](#)
 - enabled, [464](#)
 - focus, [464](#)
 - getMinimalHeightForWidth, [464](#)
 - getMinimalWidth, [464](#)
 - getPreferredHeightForWidth, [464](#)
 - getPreferredWidth, [465](#)
 - getProperty, [465](#)
 - hasFocus, [465](#)
 - horizontalStretchAffinity, [465](#)
 - hstretch, [466](#)
 - init, [466](#)
 - innerSize, [466](#)
 - insertColumn, [466](#)
 - insertRow, [467](#)
 - isAlive, [467](#)
 - isStyleClass, [467](#)
 - mayFocus, [467](#)
 - name, [467](#)
 - nameScope, [468](#)
 - OnDestroyed, [468](#)
 - OnKeyDown, [468](#)
 - OnKeyPress, [468](#)
 - OnKeyUp, [468](#)
 - OnPropertyChanged, [468](#)
 - OnResized, [469](#)
 - OnVisibilityChanged, [469](#)
 - outerSize, [469](#)
 - parentWidget, [469](#)
 - registerBusy, [469](#)
 - relayout, [469](#)
 - remove, [470](#)
 - removeColumn, [470](#)
 - removeRow, [470](#)
 - removeStyleClass, [471](#)
 - resize, [471](#)
 - setBusy, [471](#)
 - setChildren, [471](#)
 - setCols, [472](#)
 - setDoStandaloneResizing, [472](#)
 - setEnabled, [472](#)
 - setHorizontalStretchAffinity, [472](#)
 - setHstretch, [473](#)
 - setMayFocus, [473](#)
 - setName, [473](#)
 - setProperty, [474](#)
 - setStrictHorizontalSizing, [474](#)
 - setStrictVerticalSizing, [474](#)
 - setStyle, [475](#)
 - setStyleClass, [475](#)
 - setStyleClassAssigned, [475](#)
 - setVerticalStretchAffinity, [475](#)
 - setVisibility, [477](#)
 - setVstretch, [477](#)
 - strictHorizontalSizing, [477](#)
 - strictVerticalSizing, [477](#)
 - style, [477](#)
 - styleClass, [478](#)
 - unregisterBusy, [478](#)
 - verticalStretchAffinity, [478](#)
 - visibility, [478](#)
 - vstretch, [478](#)
- clove::HtmlView, [479](#)
 - addStyleClass, [482](#)
 - bindProperty, [482](#)
 - busy, [483](#)
 - childrenWidgets, [483](#)
 - computeMinimalHeightForWidth, [483](#)
 - computeMinimalWidth, [483](#)
 - computePreferredHeightForWidth, [483](#)
 - computePreferredWidth, [484](#)
 - containingNameScope, [484](#)
 - contentRoot, [484](#)
 - declareProperty, [484](#)
 - doStandaloneResizing, [485](#)
 - doinit, [485](#)
 - doinitEarly, [485](#)
 - doresize, [485](#)
 - effectiveVisibility, [486](#)
 - effectivelyEnabled, [485](#)
 - enabled, [486](#)
 - focus, [486](#)
 - getMinimalHeightForWidth, [486](#)
 - getMinimalWidth, [486](#)
 - getPreferredHeightForWidth, [487](#)
 - getPreferredWidth, [487](#)
 - getProperty, [487](#)
 - hasFocus, [488](#)

- horizontalStretchAffinity, 488
- hstretch, 488
- init, 488
- innerSize, 488
- isAlive, 488
- isStyleClass, 489
- mayFocus, 489
- name, 489
- nameScope, 489
- OnDestroyed, 489
- OnKeyDown, 490
- OnKeyPress, 490
- OnKeyUp, 490
- OnPropertyChanged, 490
- OnResized, 490
- OnVisibilityChanged, 490
- outerSize, 491
- parentWidget, 491
- registerBusy, 491
- relayout, 491
- remove, 491
- removeStyleClass, 492
- resize, 492
- setBusy, 492
- setContentRoot, 492
- setDoStandaloneResizing, 493
- setEnabled, 493
- setHorizontalStretchAffinity, 493
- setHstretch, 493
- setMayFocus, 494
- setName, 494
- setProperty, 494
- setStrictHorizontalSizing, 495
- setStrictVerticalSizing, 495
- setStyle, 495
- setStyleClass, 495
- setStyleClassAssigned, 496
- setVerticalStretchAffinity, 496
- setVisibility, 496
- setVstretch, 497
- strictHorizontalSizing, 497
- strictVerticalSizing, 497
- style, 497
- styleClass, 497
- unregisterBusy, 498
- verticalStretchAffinity, 498
- visibility, 498
- vstretch, 498
- clove::I18N, 499
 - addString, 499
 - parseLangCode, 499
 - setLanguage, 500
- clove::Icon, 500
 - bySymbol, 501
 - byUrl, 501
 - createHtml, 501
- clove::IconView, 502
 - addStyleClass, 505
 - bindProperty, 505
 - busy, 506
 - childrenWidgets, 506
 - computeMinimalHeightForWidth, 506
 - computeMinimalWidth, 506
 - computePreferredHeightForWidth, 506
 - computePreferredWidth, 507
 - containingNameScope, 507
 - declareProperty, 507
 - doStandaloneResizing, 508
 - doinit, 507
 - doinitEarly, 507
 - doresize, 508
 - effectiveVisibility, 508
 - effectivelyEnabled, 508
 - enabled, 508
 - focus, 509
 - getMinimalHeightForWidth, 509
 - getMinimalWidth, 509
 - getPreferredHeightForWidth, 509
 - getPreferredWidth, 510
 - getProperty, 510
 - hasFocus, 510
 - horizontalStretchAffinity, 510
 - hstretch, 510
 - icon, 511
 - init, 511
 - innerSize, 511
 - isAlive, 511
 - isStyleClass, 511
 - mayFocus, 512
 - name, 512
 - nameScope, 512
 - OnDestroyed, 512
 - OnKeyDown, 512
 - OnKeyPress, 513
 - OnKeyUp, 513
 - OnPropertyChanged, 513
 - OnResized, 513
 - OnVisibilityChanged, 513
 - outerSize, 513
 - parentWidget, 514
 - registerBusy, 514
 - relayout, 514
 - remove, 514
 - removeStyleClass, 515
 - resize, 515
 - setBusy, 515
 - setDoStandaloneResizing, 515
 - setEnabled, 516
 - setHorizontalStretchAffinity, 516
 - setHstretch, 516
 - setIcon, 516
 - setMayFocus, 517
 - setName, 517
 - setProperty, 517
 - setSize, 518
 - setStrictHorizontalSizing, 518

- setStrictVerticalSizing, [518](#)
- setStyle, [518](#)
- setStyleClass, [519](#)
- setStyleClassAssigned, [519](#)
- setVerticalStretchAffinity, [519](#)
- setVisibility, [520](#)
- setVstretch, [520](#)
- size, [520](#)
- strictHorizontalSizing, [520](#)
- strictVerticalSizing, [520](#)
- style, [521](#)
- styleClass, [521](#)
- unregisterBusy, [521](#)
- verticalStretchAffinity, [521](#)
- visibility, [521](#)
- vstretch, [522](#)
- clove::ImageView, [522](#)
 - addStyleClass, [525](#)
 - bindProperty, [526](#)
 - busy, [526](#)
 - childrenWidgets, [526](#)
 - computeMinimalHeightForWidth, [526](#)
 - computeMinimalWidth, [527](#)
 - computePreferredHeightForWidth, [527](#)
 - computePreferredWidth, [527](#)
 - containingNameScope, [527](#)
 - declareProperty, [527](#)
 - doStandaloneResizing, [528](#)
 - doinit, [528](#)
 - doinitEarly, [528](#)
 - doresize, [528](#)
 - effectiveVisibility, [529](#)
 - effectivelyEnabled, [529](#)
 - enabled, [529](#)
 - focus, [529](#)
 - getMinimalHeightForWidth, [529](#)
 - getMinimalWidth, [530](#)
 - getPreferredHeightForWidth, [530](#)
 - getPreferredWidth, [530](#)
 - getProperty, [530](#)
 - hasFocus, [531](#)
 - horizontalStretchAffinity, [531](#)
 - hstretch, [531](#)
 - init, [531](#)
 - innerSize, [531](#)
 - isAlive, [532](#)
 - isStyleClass, [532](#)
 - keepAspectRatio, [532](#)
 - mayFocus, [532](#)
 - name, [532](#)
 - nameScope, [533](#)
 - OnDestroyed, [533](#)
 - OnKeyDown, [533](#)
 - OnKeyPress, [533](#)
 - OnKeyUp, [533](#)
 - OnLoaded, [533](#)
 - OnPropertyChanged, [534](#)
 - OnResized, [534](#)
 - OnVisibilityChanged, [534](#)
 - outerSize, [534](#)
 - parentWidget, [534](#)
 - registerBusy, [534](#)
 - relayout, [535](#)
 - remove, [535](#)
 - removeStyleClass, [535](#)
 - resize, [536](#)
 - setBusy, [536](#)
 - setDoStandaloneResizing, [536](#)
 - setEnabled, [536](#)
 - setHorizontalStretchAffinity, [537](#)
 - setHstretch, [537](#)
 - setKeepAspectRatio, [537](#)
 - setMayFocus, [537](#)
 - setName, [538](#)
 - setProperty, [538](#)
 - setSource, [538](#)
 - setStrictHorizontalSizing, [539](#)
 - setStrictVerticalSizing, [539](#)
 - setStyle, [539](#)
 - setStyleClass, [539](#)
 - setStyleClassAssigned, [540](#)
 - setVerticalStretchAffinity, [540](#)
 - setVisibility, [540](#)
 - setVstretch, [540](#)
 - setZoom, [541](#)
 - source, [541](#)
 - strictHorizontalSizing, [541](#)
 - strictVerticalSizing, [541](#)
 - style, [541](#)
 - styleClass, [542](#)
 - unregisterBusy, [542](#)
 - verticalStretchAffinity, [542](#)
 - visibility, [542](#)
 - vstretch, [542](#)
 - zoom, [543](#)
- clove::Label, [543](#)
 - addStyleClass, [546](#)
 - bindProperty, [547](#)
 - busy, [547](#)
 - childrenWidgets, [547](#)
 - computeMinimalHeightForWidth, [547](#)
 - computeMinimalWidth, [548](#)
 - computePreferredHeightForWidth, [548](#)
 - computePreferredWidth, [548](#)
 - containingNameScope, [548](#)
 - declareProperty, [548](#)
 - doStandaloneResizing, [549](#)
 - doinit, [549](#)
 - doinitEarly, [549](#)
 - doresize, [549](#)
 - effectiveVisibility, [550](#)
 - effectivelyEnabled, [550](#)
 - enabled, [550](#)
 - focus, [550](#)
 - focusBuddy, [550](#)
 - getMinimalHeightForWidth, [550](#)

- getMinimalWidth, [551](#)
- getPreferredHeightForWidth, [551](#)
- getPreferredWidth, [551](#)
- getProperty, [551](#)
- hasFocus, [552](#)
- horizontalStretchAffinity, [552](#)
- hstretch, [552](#)
- htmlContent, [552](#)
- init, [552](#)
- innerSize, [553](#)
- isAlive, [553](#)
- isStyleClass, [553](#)
- label, [553](#)
- mayFocus, [553](#)
- name, [553](#)
- nameScope, [554](#)
- OnDestroyed, [554](#)
- OnKeyDown, [554](#)
- OnKeyPress, [554](#)
- OnKeyUp, [554](#)
- OnPropertyChanged, [554](#)
- OnResized, [555](#)
- OnVisibilityChanged, [555](#)
- outerSize, [555](#)
- parentWidget, [555](#)
- registerBusy, [555](#)
- relayout, [555](#)
- remove, [556](#)
- removeStyleClass, [556](#)
- resize, [556](#)
- setBusy, [556](#)
- setDoStandaloneResizing, [557](#)
- setEnabled, [557](#)
- setFocusBuddy, [557](#)
- setHorizontalStretchAffinity, [558](#)
- setHstretch, [558](#)
- setHtmlContent, [558](#)
- setLabel, [558](#)
- setMayFocus, [559](#)
- setName, [559](#)
- setProperty, [559](#)
- setStrictHorizontalSizing, [560](#)
- setStrictVerticalSizing, [560](#)
- setStyle, [560](#)
- setStyleClass, [560](#)
- setStyleClassAssigned, [561](#)
- setVerticalStretchAffinity, [561](#)
- setVisibility, [561](#)
- setVstretch, [562](#)
- strictHorizontalSizing, [562](#)
- strictVerticalSizing, [562](#)
- style, [562](#)
- styleClass, [562](#)
- unregisterBusy, [563](#)
- verticalStretchAffinity, [563](#)
- visibility, [563](#)
- vstretch, [563](#)
- clove::Layout, [564](#)
- addChild, [564](#)
- children, [565](#)
- clearChilds, [565](#)
- setChildren, [565](#)
- clove::ListView, [565](#)
- addStyleClass, [570](#)
- allowChecking, [571](#)
- allowSelection, [571](#)
- alwaysAllocateExpanderSpace, [571](#)
- bindProperty, [571](#)
- busy, [571](#)
- cellGenerator, [572](#)
- checkedCells, [572](#)
- childrenWidgets, [572](#)
- collapseCell, [572](#)
- columnsResizable, [572](#)
- computeMinimalHeightForWidth, [573](#)
- computeMinimalWidth, [573](#)
- computePreferredHeightForWidth, [573](#)
- computePreferredWidth, [573](#)
- containingNameScope, [574](#)
- dataViewProcessor, [574](#)
- datasource, [574](#)
- declareProperty, [574](#)
- doStandaloneResizing, [575](#)
- doinit, [574](#)
- doinitEarly, [575](#)
- doresize, [575](#)
- editCell, [575](#)
- editOnGesture, [576](#)
- effectiveVisibility, [576](#)
- effectivelyEnabled, [576](#)
- enabled, [576](#)
- expandCell, [576](#)
- expandCellRecursive, [577](#)
- focus, [577](#)
- getMinimalHeightForWidth, [577](#)
- getMinimalWidth, [577](#)
- getPreferredHeightForWidth, [577](#)
- getPreferredWidth, [578](#)
- getProperty, [578](#)
- granularity, [578](#)
- gridVisible, [578](#)
- hasFocus, [579](#)
- headersource, [579](#)
- hideExpanders, [579](#)
- horizontalStretchAffinity, [579](#)
- hstretch, [579](#)
- init, [579](#)
- innerSize, [580](#)
- isAlive, [580](#)
- isCellChecked, [580](#)
- isCellExpanded, [580](#)
- isCellSelected, [581](#)
- isStyleClass, [581](#)
- mayFocus, [581](#)
- name, [581](#)
- nameScope, [581](#)

- nodeActivationNeedsDoubleClick, 581
- OnDestroyed, 582
- OnKeyDown, 582
- OnKeyPress, 582
- OnKeyUp, 582
- OnPropertyChanged, 582
- OnResized, 582
- OnSelectionChanged, 583
- OnVisibilityChanged, 583
- outerSize, 583
- parentWidget, 583
- registerBusy, 583
- relayout, 583
- remove, 584
- removeStyleClass, 584
- resize, 584
- rowsResizable, 584
- selectCell, 585
- selection, 585
- setAllowChecking, 585
- setAllowSelection, 585
- setAlwaysAllocateExpanderSpace, 586
- setBusy, 586
- setCellChecked, 586
- setCellGenerator, 586
- setCheckedCells, 587
- setColumnsResizable, 587
- setDataViewProcessor, 588
- setDatasource, 587
- setDoStandaloneResizing, 588
- setEditOnGesture, 588
- setEnabled, 588
- setGranularity, 589
- setGridVisible, 589
- setHeadersource, 589
- setHideExpanders, 590
- setHorizontalStretchAffinity, 590
- setHstretch, 590
- setMayFocus, 590
- setName, 591
- setNodeActivationNeedsDoubleClick, 591
- setProperty, 591
- setRowsResizable, 592
- setShowChangeMenu, 592
- setShowOnlyFirstColumn, 592
- setStrictHorizontalSizing, 592
- setStrictVerticalSizing, 593
- setStyle, 593
- setStyleClass, 593
- setStyleClassAssigned, 594
- setVerticalStretchAffinity, 594
- setVisibility, 594
- setVstretch, 594
- showChangeMenu, 595
- showOnlyFirstColumn, 595
- strictHorizontalSizing, 595
- strictVerticalSizing, 595
- style, 596
- styleClass, 596
- unregisterBusy, 596
- verticalStretchAffinity, 596
- visibility, 596
- vstretch, 597
- clove::MainView, 597
 - actions, 601
 - addStyleClass, 601
 - bindProperty, 602
 - bodyLeft, 602
 - bodyLeftViewActionLabel, 602
 - bodyRight, 602
 - bodyRightViewActionLabel, 602
 - busy, 603
 - childrenWidgets, 603
 - computeMinimalHeightForWidth, 603
 - computeMinimalWidth, 603
 - computePreferredHeightForWidth, 603
 - computePreferredWidth, 604
 - containingNameScope, 604
 - declareProperty, 604
 - doStandaloneResizing, 605
 - doinit, 604
 - doinitEarly, 605
 - doresize, 605
 - effectiveVisibility, 605
 - effectivelyEnabled, 605
 - enabled, 606
 - focus, 606
 - getMinimalHeightForWidth, 606
 - getMinimalWidth, 606
 - getPreferredHeightForWidth, 606
 - getPreferredWidth, 607
 - getProperty, 607
 - hasFocus, 607
 - head1, 607
 - head2, 608
 - headControl, 608
 - headControlWidget, 608
 - horizontalStretchAffinity, 608
 - hstretch, 608
 - icon, 608
 - init, 609
 - innerSize, 609
 - isAlive, 609
 - isStyleClass, 609
 - mayFocus, 610
 - name, 610
 - nameScope, 610
 - OnDestroyed, 610
 - OnKeyDown, 610
 - OnKeyPress, 610
 - OnKeyUp, 611
 - OnPropertyChanged, 611
 - OnResized, 611
 - OnVisibilityChanged, 611
 - outerSize, 611
 - parentWidget, 612

- registerBusy, [612](#)
- relayout, [612](#)
- remove, [612](#)
- removeStyleClass, [613](#)
- resize, [613](#)
- setActions, [613](#)
- setBodyLeft, [613](#)
- setBodyLeftViewActionLabel, [614](#)
- setBodyRight, [614](#)
- setBodyRightViewActionLabel, [614](#)
- setBusy, [614](#)
- setDoStandaloneResizing, [615](#)
- setEnabled, [615](#)
- setHead1, [615](#)
- setHead2, [616](#)
- setHeadControl, [616](#)
- setHorizontalStretchAffinity, [616](#)
- setHstretch, [616](#)
- setIcon, [617](#)
- setMayFocus, [617](#)
- setName, [617](#)
- setProperty, [618](#)
- setShowOnly, [618](#)
- setSidebar, [618](#)
- setSplitterPosition, [619](#)
- setStrictHorizontalSizing, [619](#)
- setStrictVerticalSizing, [619](#)
- setStyle, [619](#)
- setStyleClass, [620](#)
- setStyleClassAssigned, [620](#)
- setVerticalStretchAffinity, [620](#)
- setVisibility, [620](#)
- setVstretch, [621](#)
- showOnly, [621](#)
- sidebar, [621](#)
- splitterPosition, [621](#)
- strictHorizontalSizing, [621](#)
- strictVerticalSizing, [622](#)
- style, [622](#)
- styleClass, [622](#)
- switchToLeftBody, [622](#)
- switchToRightBody, [622](#)
- unregisterBusy, [622](#)
- verticalStretchAffinity, [623](#)
- visibility, [623](#)
- vstretch, [623](#)
- clove::MediaPlayer, [623](#)
 - addStyleClass, [628](#)
 - autoPlay, [628](#)
 - bindProperty, [628](#)
 - busy, [629](#)
 - childrenWidgets, [629](#)
 - computeMinimalHeightForWidth, [629](#)
 - computeMinimalWidth, [629](#)
 - computePreferredHeightForWidth, [629](#)
 - computePreferredWidth, [630](#)
 - containingNameScope, [630](#)
 - currentMediaPosition, [630](#)
 - declareProperty, [630](#)
 - doStandaloneResizing, [631](#)
 - doinit, [630](#)
 - doinitEarly, [631](#)
 - doresize, [631](#)
 - duration, [631](#)
 - effectiveVisibility, [632](#)
 - effectivelyEnabled, [631](#)
 - enabled, [632](#)
 - focus, [632](#)
 - getMinimalHeightForWidth, [632](#)
 - getMinimalWidth, [632](#)
 - getPreferredHeightForWidth, [633](#)
 - getPreferredWidth, [633](#)
 - getProperty, [633](#)
 - hasEnded, [634](#)
 - hasFocus, [634](#)
 - horizontalStretchAffinity, [634](#)
 - hstretch, [634](#)
 - init, [634](#)
 - innerSize, [635](#)
 - isAlive, [635](#)
 - isPaused, [635](#)
 - isStyleClass, [635](#)
 - loop, [635](#)
 - mayFocus, [635](#)
 - muted, [636](#)
 - name, [636](#)
 - nameScope, [636](#)
 - nativeNode, [636](#)
 - OnDestroyed, [636](#)
 - OnDurationChanged, [636](#)
 - OnHasEnded, [637](#)
 - OnIsPaused, [637](#)
 - OnIsPlaying, [637](#)
 - OnKeyDown, [637](#)
 - OnKeyPress, [637](#)
 - OnKeyUp, [638](#)
 - OnLoadingAborted, [638](#)
 - OnPropertyChanged, [638](#)
 - OnReadyForPlayback, [638](#)
 - OnResized, [638](#)
 - OnVisibilityChanged, [638](#)
 - outerSize, [639](#)
 - parentWidget, [639](#)
 - pause, [639](#)
 - play, [639](#)
 - preload, [639](#)
 - registerBusy, [639](#)
 - relayout, [640](#)
 - remove, [640](#)
 - removeStyleClass, [640](#)
 - resize, [641](#)
 - setAutoPlay, [641](#)
 - setBusy, [641](#)
 - setCurrentMediaPosition, [641](#)
 - setDoStandaloneResizing, [642](#)
 - setEnabled, [642](#)

- setHorizontalStretchAffinity, 642
- setHstretch, 642
- setLoop, 643
- setMayFocus, 643
- setMuted, 643
- setName, 643
- setPreload, 644
- setProperty, 644
- setShowControls, 644
- setSource, 645
- setStrictHorizontalSizing, 645
- setStrictVerticalSizing, 645
- setStyle, 645
- setStyleClass, 646
- setStyleClassAssigned, 646
- setVerticalStretchAffinity, 646
- setVisibility, 647
- setVolume, 647
- setVstretch, 647
- showControls, 647
- source, 648
- strictHorizontalSizing, 648
- strictVerticalSizing, 648
- style, 648
- styleClass, 648
- unregisterBusy, 648
- verticalStretchAffinity, 649
- visibility, 649
- volume, 649
- vstretch, 649
- clove::MenuBar, 650
 - actions, 653
 - addStyleClass, 653
 - bindProperty, 654
 - busy, 654
 - childrenWidgets, 654
 - computeMinimalHeightForWidth, 654
 - computeMinimalWidth, 655
 - computePreferredHeightForWidth, 655
 - computePreferredWidth, 655
 - containingNameScope, 655
 - declareProperty, 655
 - doStandaloneResizing, 656
 - doinit, 656
 - doinitEarly, 656
 - doresize, 656
 - effectiveVisibility, 657
 - effectivelyEnabled, 657
 - enabled, 657
 - focus, 657
 - getMinimalHeightForWidth, 657
 - getMinimalWidth, 658
 - getPreferredHeightForWidth, 658
 - getPreferredWidth, 658
 - getProperty, 658
 - hasFocus, 659
 - horizontalStretchAffinity, 659
 - hstretch, 659
 - init, 659
 - innerSize, 659
 - isAlive, 660
 - isStyleClass, 660
 - mayFocus, 660
 - name, 660
 - nameScope, 660
 - OnActionTriggered, 661
 - OnBeforeSubactionsExpanded, 661
 - OnDestroyed, 661
 - OnKeyDown, 661
 - OnKeyPress, 661
 - OnKeyUp, 661
 - OnPropertyChanged, 662
 - OnResized, 662
 - OnVisibilityChanged, 662
 - outerSize, 662
 - parentWidget, 662
 - registerBusy, 662
 - relayout, 663
 - remove, 663
 - removeStyleClass, 663
 - resize, 664
 - setActions, 664
 - setBusy, 664
 - setDoStandaloneResizing, 664
 - setEnabled, 665
 - setHorizontalStretchAffinity, 665
 - setHstretch, 665
 - setMayFocus, 665
 - setName, 666
 - setProperty, 666
 - setStrictHorizontalSizing, 666
 - setStrictVerticalSizing, 667
 - setStyle, 667
 - setStyleClass, 667
 - setStyleClassAssigned, 667
 - setVerticalStretchAffinity, 668
 - setVisibility, 668
 - setVstretch, 668
 - strictHorizontalSizing, 668
 - strictVerticalSizing, 669
 - style, 669
 - styleClass, 669
 - unregisterBusy, 669
 - verticalStretchAffinity, 669
 - visibility, 670
 - vstretch, 670
- clove::ModalPanel, 670
 - addStyleClass, 674
 - bindProperty, 674
 - busy, 674
 - childrenWidgets, 674
 - computeMinimalHeightForWidth, 675
 - computeMinimalWidth, 675
 - computePreferredHeightForWidth, 675
 - computePreferredWidth, 675
 - containingNameScope, 676

declareProperty, 676
doStandaloneResizing, 677
doInit, 676
doInitEarly, 676
doResize, 676
effectiveVisibility, 677
effectivelyEnabled, 677
enabled, 677
focus, 677
fullyTransparent, 677
getMinimalHeightForWidth, 678
getMinimalWidth, 679
getPreferredHeightForWidth, 679
getPreferredWidth, 679
getProperty, 679
hasFocus, 680
horizontalStretchAffinity, 680
hstretch, 680
init, 680
innerSize, 680
isAlive, 681
isStyleClass, 681
mayFocus, 681
name, 681
nameScope, 681
OnClicked, 682
OnDestroyed, 682
OnKeyDown, 682
OnKeyPress, 682
OnKeyUp, 682
OnPropertyChanged, 682
OnResized, 683
OnVisibilityChanged, 683
outerSize, 683
parentWidget, 683
registerBusy, 683
relayout, 683
remove, 684
removeStyleClass, 684
resize, 684
setBusy, 684
setDoStandaloneResizing, 685
setEnabled, 685
setFullyTransparent, 685
setHorizontalStretchAffinity, 686
setHstretch, 686
setMayFocus, 686
setName, 686
setProperty, 687
setStrictHorizontalSizing, 687
setStrictVerticalSizing, 687
setStyle, 688
setStyleClass, 688
setStyleClassAssigned, 688
setVerticalStretchAffinity, 688
setVisibility, 689
setVstretch, 689
strictHorizontalSizing, 689
strictVerticalSizing, 689
style, 690
styleClass, 690
unregisterBusy, 690
verticalStretchAffinity, 690
visibility, 690
vstretch, 691
clove::MultilineEditBox, 691
addStyleClass, 695
autocompletionFilter, 695
autocompletionItems, 695
autocompletionOpenForNoText, 695
bindProperty, 696
busy, 696
childrenWidgets, 696
computeMinimalHeightForWidth, 696
computeMinimalWidth, 697
computePreferredHeightForWidth, 697
computePreferredWidth, 697
containingNameScope, 697
declareProperty, 697
doStandaloneResizing, 698
doInit, 698
doInitEarly, 698
doResize, 698
effectiveVisibility, 699
effectivelyEnabled, 699
enabled, 699
focus, 699
getMinimalHeightForWidth, 699
getMinimalWidth, 700
getPreferredHeightForWidth, 700
getPreferredWidth, 700
getProperty, 700
hasFocus, 701
hintText, 701
horizontalStretchAffinity, 701
hstretch, 701
init, 701
innerSize, 702
isAlive, 702
isStyleClass, 702
mayFocus, 702
name, 702
nameScope, 702
OnChanged, 703
OnDestroyed, 703
OnKeyDown, 703
OnKeyPress, 703
OnKeyUp, 703
OnPopupTextSelected, 703
OnPropertyChanged, 704
OnResized, 704
OnVisibilityChanged, 704
outerSize, 704
parentWidget, 704
readOnly, 704
registerBusy, 705

- relayout, 705
- remove, 705
- removeStyleClass, 705
- resize, 706
- setAutocompletionFilter, 706
- setAutocompletionItems, 706
- setAutocompletionOpenForNoText, 706
- setBusy, 707
- setDoStandaloneResizing, 707
- setEnabled, 707
- setHintText, 707
- setHorizontalStretchAffinity, 708
- setHstretch, 708
- setMayFocus, 708
- setName, 709
- setProperty, 709
- setReadOnly, 709
- setStrictHorizontalSizing, 710
- setStrictVerticalSizing, 710
- setStyle, 710
- setStyleClass, 710
- setStyleClassAssigned, 711
- setText, 711
- setTextSelection, 711
- setVerticalStretchAffinity, 712
- setVisibility, 712
- setVstretch, 712
- strictHorizontalSizing, 712
- strictVerticalSizing, 712
- style, 713
- styleClass, 713
- text, 713
- textSelection, 713
- unregisterBusy, 713
- verticalStretchAffinity, 714
- visibility, 714
- vstretch, 714
- clove::NameScope, 714
 - addChildNameScope, 715
 - getByName, 715
- clove::NativeDatasource, 716
 - addColumn, 718
 - appendRow, 718
 - changeValue, 718
 - columnCount, 719
 - columnHeadersVisible, 719
 - getColumnHeader, 719
 - getMetadata, 719
 - getRowHeader, 720
 - getValue, 720
 - insertColumn, 720
 - insertRow, 721
 - NativeDatasource, 717
 - OnDataInsert, 721
 - OnDataRemove, 721
 - OnDataUpdate, 721
 - OnHeaderDataInsert, 721
 - OnHeaderDataRemove, 722
 - OnHeaderDataUpdate, 722
 - OnHeaderVisibilityUpdated, 722
 - parent, 722
 - removeColumn, 722
 - removeRow, 723
 - root, 723
 - rowCount, 723
 - rowHeadersVisible, 723
 - setColumnHeader, 724
 - setMetadata, 724
 - setRootValue, 724
 - setRowHeader, 725
 - setValue, 725
 - valuePointer, 725
 - valuePointerNavigateInDepth, 726
 - valuePointerToNativeNode, 726
- clove::NativeDatasourceNode, 727
 - addColumn, 727
 - addRow, 729
 - columnCount, 729
 - getChild, 729
 - getMetadata, 729
 - getValue, 729
 - insertColumn, 730
 - insertRow, 730
 - removeColumn, 730
 - removeRow, 731
 - rowCount, 731
 - setMetadata, 731
 - setValue, 731
 - toColumn, 732
 - toRow, 732
 - valuePointer, 732
- clove::NotificationController, 732
 - notify, 733
- clove::NumericEditBox, 733
 - addStyleClass, 738
 - autocompletionFilter, 738
 - autocompletionItems, 738
 - autocompletionOpenForNoText, 738
 - bindProperty, 738
 - busy, 739
 - childrenWidgets, 739
 - computeMinimalHeightForWidth, 739
 - computeMinimalWidth, 739
 - computePreferredHeightForWidth, 739
 - computePreferredWidth, 740
 - containingNameScope, 740
 - declareProperty, 740
 - doStandaloneResizing, 741
 - doinit, 740
 - doinitEarly, 741
 - doresize, 741
 - effectiveVisibility, 741
 - effectivelyEnabled, 741
 - enabled, 742
 - focus, 742
 - getMinimalHeightForWidth, 742

getMinimalWidth, 742
getPreferredHeightForWidth, 742
getPreferredWidth, 743
getProperty, 743
hasFocus, 743
hintText, 743
horizontalStretchAffinity, 744
hstretch, 744
init, 744
innerSize, 744
isAlive, 744
isStyleClass, 745
max, 745
mayFocus, 745
min, 745
name, 745
nameScope, 746
OnChanged, 746
OnDestroyed, 746
OnKeyDown, 746
OnKeyPress, 746
OnKeyUp, 746
OnPopupTextSelected, 747
OnPropertyChanged, 747
OnResized, 747
OnVisibilityChanged, 747
outerSize, 747
parentWidget, 747
readOnly, 748
registerBusy, 748
relayout, 748
remove, 748
removeStyleClass, 749
resize, 749
setAutocompletionFilter, 749
setAutocompletionItems, 749
setAutocompletionOpenForNoText, 750
setBusy, 750
setDoStandaloneResizing, 750
setEnabled, 750
setHintText, 751
setHorizontalStretchAffinity, 751
setHstretch, 751
setMax, 752
setMayFocus, 752
setMin, 752
setName, 752
setProperty, 753
setReadOnly, 753
setStepsize, 753
setStrictHorizontalSizing, 754
setStrictVerticalSizing, 754
setStyle, 754
setStyleClass, 754
setStyleClassAssigned, 755
setText, 755
setTextSelection, 755
setValue, 756
setVerticalStretchAffinity, 756
setVisibility, 756
setVstretch, 756
stepsize, 757
strictHorizontalSizing, 757
strictVerticalSizing, 757
style, 757
styleClass, 757
text, 758
textSelection, 758
unregisterBusy, 758
value, 758
verticalStretchAffinity, 758
visibility, 759
vstretch, 759
clove::PasswordEditBox, 759
 addStyleClass, 763
 autocompletionFilter, 763
 autocompletionItems, 764
 autocompletionOpenForNoText, 764
 bindProperty, 764
 busy, 764
 childrenWidgets, 764
 computeMinimalHeightForWidth, 765
 computeMinimalWidth, 765
 computePreferredHeightForWidth, 765
 computePreferredWidth, 765
 containingNameScope, 766
 declareProperty, 766
 doStandaloneResizing, 767
 doinit, 766
 doinitEarly, 766
 doresize, 766
 effectiveVisibility, 767
 effectivelyEnabled, 767
 enabled, 767
 focus, 767
 getMinimalHeightForWidth, 767
 getMinimalWidth, 768
 getPreferredHeightForWidth, 768
 getPreferredWidth, 768
 getProperty, 768
 hasFocus, 769
 hintText, 769
 horizontalStretchAffinity, 769
 hstretch, 769
 init, 769
 innerSize, 770
 isAlive, 770
 isStyleClass, 770
 mayFocus, 770
 name, 770
 nameScope, 770
 OnChanged, 771
 OnDestroyed, 771
 OnKeyDown, 771
 OnKeyPress, 771
 OnKeyUp, 771

- OnPopupTextSelected, [771](#)
- OnPropertyChanged, [772](#)
- OnResized, [772](#)
- OnVisibilityChanged, [772](#)
- outerSize, [772](#)
- parentWidget, [772](#)
- readOnly, [772](#)
- registerBusy, [773](#)
- relayout, [773](#)
- remove, [773](#)
- removeStyleClass, [773](#)
- resize, [774](#)
- setAutocompletionFilter, [774](#)
- setAutocompletionItems, [774](#)
- setAutocompletionOpenForNoText, [774](#)
- setBusy, [775](#)
- setDoStandaloneResizing, [775](#)
- setEnabled, [775](#)
- setHintText, [775](#)
- setHorizontalStretchAffinity, [776](#)
- setHstretch, [776](#)
- setMayFocus, [776](#)
- setName, [777](#)
- setProperty, [777](#)
- setReadOnly, [777](#)
- setStrictHorizontalSizing, [778](#)
- setStrictVerticalSizing, [778](#)
- setStyle, [778](#)
- setStyleClass, [778](#)
- setStyleClassAssigned, [779](#)
- setText, [779](#)
- setTextSelection, [779](#)
- setVerticalStretchAffinity, [780](#)
- setVisibility, [780](#)
- setVstretch, [780](#)
- strictHorizontalSizing, [780](#)
- strictVerticalSizing, [780](#)
- style, [781](#)
- styleClass, [781](#)
- text, [781](#)
- textSelection, [781](#)
- unregisterBusy, [781](#)
- verticalStretchAffinity, [782](#)
- visibility, [782](#)
- vstretch, [782](#)
- clove::PopupMenu, [782](#)
 - actions, [786](#)
 - addStyleClass, [786](#)
 - bindProperty, [787](#)
 - busy, [787](#)
 - childrenWidgets, [787](#)
 - computeMinimalHeightForWidth, [787](#)
 - computeMinimalWidth, [788](#)
 - computePreferredHeightForWidth, [788](#)
 - computePreferredWidth, [788](#)
 - containingNameScope, [788](#)
 - declareProperty, [788](#)
 - doStandaloneResizing, [789](#)
 - doinit, [789](#)
 - doinitEarly, [789](#)
 - doresize, [789](#)
 - effectiveVisibility, [790](#)
 - effectivelyEnabled, [790](#)
 - enabled, [790](#)
 - expanderIndicatorDirection, [790](#)
 - focus, [790](#)
 - getMinimalHeightForWidth, [790](#)
 - getMinimalWidth, [791](#)
 - getPreferredHeightForWidth, [791](#)
 - getPreferredWidth, [791](#)
 - getProperty, [791](#)
 - hasFocus, [792](#)
 - horizontalStretchAffinity, [792](#)
 - hstretch, [792](#)
 - init, [792](#)
 - innerSize, [792](#)
 - isAlive, [793](#)
 - isStyleClass, [793](#)
 - mayFocus, [793](#)
 - name, [793](#)
 - nameScope, [793](#)
 - OnActionTriggered, [794](#)
 - OnBeforeSubactionsExpanded, [794](#)
 - OnDestroyed, [794](#)
 - OnKeyDown, [794](#)
 - OnKeyPress, [794](#)
 - OnKeyUp, [794](#)
 - OnPropertyChanged, [795](#)
 - OnResized, [795](#)
 - OnVisibilityChanged, [795](#)
 - outerSize, [795](#)
 - parentWidget, [795](#)
 - registerBusy, [795](#)
 - relayout, [796](#)
 - remove, [796](#)
 - removeStyleClass, [796](#)
 - resize, [797](#)
 - setActions, [797](#)
 - setBusy, [797](#)
 - setDoStandaloneResizing, [797](#)
 - setEnabled, [798](#)
 - setExpanderIndicatorDirection, [798](#)
 - setHorizontalStretchAffinity, [798](#)
 - setHstretch, [798](#)
 - setMayFocus, [799](#)
 - setName, [799](#)
 - setProperty, [799](#)
 - setStrictHorizontalSizing, [800](#)
 - setStrictVerticalSizing, [800](#)
 - setStyle, [800](#)
 - setStyleClass, [800](#)
 - setStyleClassAssigned, [801](#)
 - setVerticalStretchAffinity, [801](#)
 - setVisibility, [801](#)
 - setVstretch, [801](#)
 - show, [802](#)

- strictHorizontalSizing, [802](#)
- strictVerticalSizing, [802](#)
- style, [802](#)
- styleClass, [803](#)
- unregisterBusy, [803](#)
- verticalStretchAffinity, [803](#)
- visibility, [803](#)
- vstretch, [803](#)
- clove::PopupMenuButton, [804](#)
 - actions, [807](#)
 - addStyleClass, [808](#)
 - bindProperty, [808](#)
 - busy, [808](#)
 - checkable, [808](#)
 - checked, [808](#)
 - childrenWidgets, [809](#)
 - computeMinimalHeightForWidth, [809](#)
 - computeMinimalWidth, [809](#)
 - computePreferredHeightForWidth, [809](#)
 - computePreferredWidth, [810](#)
 - containingNameScope, [810](#)
 - declareProperty, [810](#)
 - doStandaloneResizing, [811](#)
 - doinit, [810](#)
 - doinitEarly, [810](#)
 - doresize, [811](#)
 - effectiveVisibility, [811](#)
 - effectivelyEnabled, [811](#)
 - enabled, [811](#)
 - focus, [812](#)
 - getMinimalHeightForWidth, [812](#)
 - getMinimalWidth, [812](#)
 - getPreferredHeightForWidth, [812](#)
 - getPreferredWidth, [813](#)
 - getProperty, [813](#)
 - hasFocus, [813](#)
 - horizontalStretchAffinity, [813](#)
 - hstretch, [813](#)
 - icon, [814](#)
 - init, [814](#)
 - innerSize, [814](#)
 - isAlive, [814](#)
 - isStyleClass, [814](#)
 - label, [815](#)
 - mayFocus, [815](#)
 - name, [815](#)
 - nameScope, [815](#)
 - OnActionTriggered, [815](#)
 - OnClicked, [816](#)
 - OnDestroyed, [816](#)
 - OnKeyDown, [816](#)
 - OnKeyPress, [816](#)
 - OnKeyUp, [816](#)
 - OnPropertyChanged, [816](#)
 - OnResized, [817](#)
 - OnVisibilityChanged, [817](#)
 - outerSize, [817](#)
 - parentWidget, [817](#)
 - registerBusy, [817](#)
 - layout, [817](#)
 - remove, [818](#)
 - removeStyleClass, [818](#)
 - resize, [818](#)
 - setActions, [818](#)
 - setBusy, [819](#)
 - setCheckable, [819](#)
 - setChecked, [819](#)
 - setDoStandaloneResizing, [820](#)
 - setEnabled, [820](#)
 - setHorizontalStretchAffinity, [820](#)
 - setHstretch, [820](#)
 - setIcon, [821](#)
 - setLabel, [821](#)
 - setMayFocus, [821](#)
 - setName, [822](#)
 - setProperty, [822](#)
 - setStrictHorizontalSizing, [822](#)
 - setStrictVerticalSizing, [823](#)
 - setStyle, [823](#)
 - setStyleClass, [823](#)
 - setStyleClassAssigned, [823](#)
 - setVerticalStretchAffinity, [824](#)
 - setVisibility, [824](#)
 - setVstretch, [824](#)
 - strictHorizontalSizing, [824](#)
 - strictVerticalSizing, [825](#)
 - style, [825](#)
 - styleClass, [825](#)
 - unregisterBusy, [825](#)
 - verticalStretchAffinity, [825](#)
 - visibility, [826](#)
 - vstretch, [826](#)
- clove::ProgressBar, [826](#)
 - addStyleClass, [830](#)
 - bindProperty, [830](#)
 - busy, [830](#)
 - childrenWidgets, [831](#)
 - computeMinimalHeightForWidth, [831](#)
 - computeMinimalWidth, [831](#)
 - computePreferredHeightForWidth, [831](#)
 - computePreferredWidth, [832](#)
 - containingNameScope, [832](#)
 - declareProperty, [832](#)
 - doStandaloneResizing, [833](#)
 - doinit, [832](#)
 - doinitEarly, [832](#)
 - doresize, [833](#)
 - effectiveVisibility, [833](#)
 - effectivelyEnabled, [833](#)
 - enabled, [833](#)
 - focus, [834](#)
 - getMinimalHeightForWidth, [834](#)
 - getMinimalWidth, [834](#)
 - getPreferredHeightForWidth, [834](#)
 - getPreferredWidth, [835](#)
 - getProperty, [835](#)

- hasFocus, 835
- horizontalStretchAffinity, 835
- hstretch, 835
- init, 836
- innerSize, 836
- isAlive, 836
- isStyleClass, 836
- label, 837
- labelFunction, 837
- mayFocus, 837
- name, 837
- nameScope, 837
- OnDestroyed, 837
- OnKeyDown, 838
- OnKeyPress, 838
- OnKeyUp, 838
- OnPropertyChanged, 838
- OnResized, 838
- OnVisibilityChanged, 838
- orientation, 839
- outerSize, 839
- parentWidget, 839
- registerBusy, 839
- relayout, 839
- remove, 839
- removeStyleClass, 840
- resize, 840
- setBusy, 840
- setDoStandaloneResizing, 840
- setEnabled, 841
- setHorizontalStretchAffinity, 841
- setHstretch, 841
- setLabel, 841
- setLabelFunction, 842
- setMayFocus, 842
- setName, 842
- setOrientation, 843
- setProperty, 843
- setStrictHorizontalSizing, 843
- setStrictVerticalSizing, 844
- setStyle, 844
- setStyleClass, 844
- setStyleClassAssigned, 844
- setValue, 845
- setVerticalStretchAffinity, 845
- setVisibility, 845
- setVstretch, 845
- strictHorizontalSizing, 846
- strictVerticalSizing, 846
- style, 846
- styleClass, 846
- unregisterBusy, 846
- value, 847
- verticalStretchAffinity, 847
- visibility, 847
- vstretch, 847
- clove::ProxyDatasource, 848
- changeValue, 849
- columnCount, 849
- columnHeadersVisible, 850
- getColumnHeader, 850
- getMetadata, 850
- getRowHeader, 850
- getValue, 851
- nativePointerToProxyPointer, 851
- OnDataInsert, 851
- OnDataRemove, 851
- OnDataUpdate, 852
- OnHeaderDataInsert, 852
- OnHeaderDataRemove, 852
- OnHeaderDataUpdate, 852
- OnHeaderVisibilityUpdated, 852
- parent, 852
- proxyPointerToNativePointer, 853
- refresh, 853
- rowCount, 853
- rowHeadersVisible, 853
- setDatasource, 854
- valuePointer, 854
- valuePointerNavigateInDepth, 854
- clove::RadioButton, 855
- addStyleClass, 858
- bindProperty, 859
- busy, 859
- checked, 859
- childrenWidgets, 859
- computeMinimalHeightForWidth, 859
- computeMinimalWidth, 860
- computePreferredHeightForWidth, 860
- computePreferredWidth, 860
- containingNameScope, 860
- declareProperty, 860
- doStandaloneResizing, 861
- doinit, 861
- doinitEarly, 861
- doresize, 861
- effectiveVisibility, 862
- effectivelyEnabled, 862
- enabled, 862
- focus, 862
- getMinimalHeightForWidth, 862
- getMinimalWidth, 863
- getPreferredHeightForWidth, 863
- getPreferredWidth, 863
- getProperty, 863
- group, 864
- groupValue, 864
- hasFocus, 864
- horizontalStretchAffinity, 864
- hstretch, 864
- init, 864
- innerSize, 865
- isAlive, 865
- isStyleClass, 865
- label, 865
- mayFocus, 865

- name, 865
- nameScope, 866
- OnChanged, 866
- OnClicked, 866
- OnDestroyed, 866
- OnKeyDown, 866
- OnKeyPress, 866
- OnKeyUp, 867
- OnPropertyChanged, 867
- OnResized, 867
- OnVisibilityChanged, 867
- outerSize, 867
- parentWidget, 868
- registerBusy, 868
- relayout, 868
- remove, 868
- removeStyleClass, 869
- resize, 869
- setBusy, 869
- setChecked, 869
- setDoStandaloneResizing, 870
- setEnabled, 870
- setGroup, 870
- setGroupValue, 870
- setHorizontalStretchAffinity, 871
- setHstretch, 871
- setLabel, 871
- setMayFocus, 872
- setName, 872
- setProperty, 872
- setStrictHorizontalSizing, 873
- setStrictVerticalSizing, 873
- setStyle, 873
- setStyleClass, 873
- setStyleClassAssigned, 874
- setValueDef, 874
- setVerticalStretchAffinity, 874
- setVisibility, 874
- setVstretch, 875
- strictHorizontalSizing, 875
- strictVerticalSizing, 875
- style, 875
- styleClass, 875
- unregisterBusy, 876
- valueDef, 876
- verticalStretchAffinity, 876
- visibility, 876
- vstretch, 876
- clove::RadioGroup, 877
 - getGroupByPublicName, 878
 - OnChanged, 878
 - RadioGroup, 877
 - select, 878
 - selectByIndex, 878
 - selectByValue, 879
 - selected, 879
 - selectedIndex, 879
 - selectedValue, 879
 - unselectAll, 879
- clove::RawResizeSplitter, 880
 - addStyleClass, 883
 - bindProperty, 883
 - busy, 884
 - childrenWidgets, 884
 - computeMinimalHeightForWidth, 884
 - computeMinimalWidth, 884
 - computePreferredHeightForWidth, 884
 - computePreferredWidth, 885
 - containingNameScope, 885
 - declareProperty, 885
 - doStandaloneResizing, 886
 - doinit, 885
 - doinitEarly, 886
 - doresize, 886
 - effectiveVisibility, 886
 - effectivelyEnabled, 886
 - enabled, 887
 - focus, 887
 - getMinimalHeightForWidth, 887
 - getMinimalWidth, 887
 - getPreferredHeightForWidth, 887
 - getPreferredWidth, 888
 - getProperty, 888
 - hasFocus, 888
 - horizontalStretchAffinity, 888
 - hstretch, 889
 - init, 889
 - innerSize, 889
 - isAlive, 889
 - isStyleClass, 889
 - mayFocus, 890
 - name, 890
 - nameScope, 890
 - OnDestroyed, 890
 - OnKeyDown, 890
 - OnKeyPress, 891
 - OnKeyUp, 891
 - OnMove, 891
 - OnMoveBegin, 891
 - OnMoveEnd, 891
 - OnPropertyChanged, 891
 - OnResized, 892
 - OnVisibilityChanged, 892
 - outerSize, 892
 - parentWidget, 892
 - registerBusy, 892
 - relayout, 892
 - remove, 893
 - removeStyleClass, 893
 - resize, 893
 - setBusy, 893
 - setDoStandaloneResizing, 894
 - setEnabled, 894
 - setHorizontalStretchAffinity, 894
 - setHstretch, 895
 - setMayFocus, 895

- setName, 895
- setProperty, 895
- setStrictHorizontalSizing, 896
- setStrictVerticalSizing, 896
- setStyle, 896
- setStyleClass, 897
- setStyleClassAssigned, 897
- setVerticalStretchAffinity, 897
- setVisibility, 897
- setVstretch, 898
- strictHorizontalSizing, 898
- strictVerticalSizing, 898
- style, 898
- styleClass, 898
- unregisterBusy, 899
- verticalStretchAffinity, 899
- visibility, 899
- vstretch, 899
- clove::ResizeSplitter, 900
 - addBodyWidget, 903
 - addStyleClass, 904
 - bindProperty, 904
 - body, 904
 - bodyWidgets, 904
 - busy, 904
 - childrenWidgets, 905
 - computeMinimalHeightForWidth, 905
 - computeMinimalWidth, 905
 - computePreferredHeightForWidth, 905
 - computePreferredWidth, 906
 - containingNameScope, 906
 - declareProperty, 906
 - doStandaloneResizing, 907
 - doinit, 906
 - doinitEarly, 906
 - doresize, 907
 - effectiveVisibility, 907
 - effectivelyEnabled, 907
 - enabled, 907
 - focus, 908
 - getMinimalHeightForWidth, 908
 - getMinimalWidth, 908
 - getPreferredHeightForWidth, 908
 - getPreferredWidth, 909
 - getProperty, 909
 - hasFocus, 909
 - horizontalStretchAffinity, 909
 - hstretch, 909
 - init, 910
 - innerSize, 910
 - insertBodyWidget, 910
 - isAlive, 910
 - isStyleClass, 911
 - mayFocus, 911
 - name, 911
 - nameScope, 911
 - OnDestroyed, 911
 - OnKeyDown, 912
 - OnKeyPress, 912
 - OnKeyUp, 912
 - OnPropertyChanged, 912
 - OnResized, 912
 - OnVisibilityChanged, 912
 - orientation, 913
 - outerSize, 913
 - parentWidget, 913
 - registerBusy, 913
 - relayout, 913
 - remove, 913
 - removeStyleClass, 914
 - resize, 914
 - setBody, 914
 - setBodyWidget, 914
 - setBusy, 916
 - setDoStandaloneResizing, 916
 - setEnabled, 916
 - setHorizontalStretchAffinity, 916
 - setHstretch, 917
 - setMayFocus, 917
 - setName, 917
 - setOrientation, 918
 - setProperty, 918
 - setShowOnly, 918
 - setSizeFractions, 919
 - setSplitterVisibility, 919
 - setSplitterWidth, 919
 - setStrictHorizontalSizing, 919
 - setStrictVerticalSizing, 920
 - setStyle, 920
 - setStyleClass, 920
 - setStyleClassAssigned, 920
 - setVerticalStretchAffinity, 921
 - setVisibility, 921
 - setVstretch, 921
 - showOnly, 922
 - sizeFractions, 922
 - splitterVisibility, 922
 - splitterWidth, 922
 - strictHorizontalSizing, 922
 - strictVerticalSizing, 922
 - style, 923
 - styleClass, 923
 - unregisterBusy, 923
 - verticalStretchAffinity, 923
 - visibility, 923
 - vstretch, 924
- clove::RichEdit, 924
 - addStyleClass, 927
 - bindProperty, 928
 - busy, 928
 - childrenWidgets, 928
 - computeMinimalHeightForWidth, 928
 - computeMinimalWidth, 929
 - computePreferredHeightForWidth, 929
 - computePreferredWidth, 929
 - containingNameScope, 929

- declareProperty, 929
- doStandaloneResizing, 930
- doinit, 930
- doinitEarly, 930
- doresize, 930
- effectiveVisibility, 931
- effectivelyEnabled, 931
- enabled, 931
- focus, 931
- getMinimalHeightForWidth, 931
- getMinimalWidth, 932
- getPreferredHeightForWidth, 932
- getPreferredWidth, 932
- getProperty, 932
- hasFocus, 933
- horizontalStretchAffinity, 933
- hstretch, 933
- htmlContent, 933
- init, 933
- innerSize, 934
- isAlive, 934
- isStyleClass, 934
- mayFocus, 934
- name, 934
- nameScope, 934
- OnDestroyed, 935
- OnKeyDown, 935
- OnKeyPress, 935
- OnKeyUp, 935
- OnPropertyChanged, 935
- OnResized, 935
- OnVisibilityChanged, 936
- outerSize, 936
- parentWidget, 936
- registerBusy, 936
- relayout, 936
- remove, 936
- removeStyleClass, 937
- resize, 937
- richeditbox, 937
- setBusy, 937
- setDoStandaloneResizing, 938
- setEnabled, 938
- setHorizontalStretchAffinity, 938
- setHstretch, 938
- setHtmlContent, 939
- setMayFocus, 939
- setName, 939
- setProperty, 939
- setStrictHorizontalSizing, 940
- setStrictVerticalSizing, 940
- setStyle, 940
- setStyleClass, 941
- setStyleClassAssigned, 941
- setVerticalStretchAffinity, 941
- setVisibility, 941
- setVstretch, 942
- strictHorizontalSizing, 942
- strictVerticalSizing, 942
- style, 942
- styleClass, 942
- unregisterBusy, 943
- verticalStretchAffinity, 943
- visibility, 943
- vstretch, 943
- clove::RichEditBox, 944
 - addStyleClass, 947
 - bindProperty, 948
 - busy, 948
 - childrenWidgets, 948
 - computeMinimalHeightForWidth, 948
 - computeMinimalWidth, 949
 - computePreferredHeightForWidth, 949
 - computePreferredWidth, 949
 - containingNameScope, 949
 - contentHtmlNode, 949
 - declareProperty, 950
 - doStandaloneResizing, 951
 - doinit, 950
 - doinitEarly, 950
 - doresize, 950
 - effectiveVisibility, 951
 - effectivelyEnabled, 951
 - enabled, 951
 - focus, 951
 - getMinimalHeightForWidth, 951
 - getMinimalWidth, 952
 - getPreferredHeightForWidth, 952
 - getPreferredWidth, 952
 - getProperty, 952
 - hasFocus, 953
 - horizontalStretchAffinity, 953
 - hstretch, 953
 - htmlContent, 953
 - init, 953
 - innerSize, 954
 - isAlive, 954
 - isStyleClass, 954
 - mayFocus, 954
 - name, 954
 - nameScope, 954
 - OnDestroyed, 955
 - OnKeyDown, 955
 - OnKeyPress, 955
 - OnKeyUp, 955
 - OnPropertyChanged, 955
 - OnResized, 955
 - OnVisibilityChanged, 956
 - outerSize, 956
 - parentWidget, 956
 - registerBusy, 956
 - relayout, 956
 - remove, 956
 - removeStyleClass, 957
 - resize, 957
 - selection, 957

- selectionDecreaseFontSize, [957](#)
- selectionIncreaseFontSize, [957](#)
- selectionInsertList, [958](#)
- selectionSetBackgroundColor, [958](#)
- selectionSetForegroundColor, [958](#)
- setBusy, [959](#)
- setDoStandaloneResizing, [959](#)
- setEnabled, [959](#)
- setHorizontalStretchAffinity, [959](#)
- setHstretch, [960](#)
- setHtmlContent, [960](#)
- setMayFocus, [960](#)
- setName, [960](#)
- setProperty, [961](#)
- setSelection, [961](#)
- setStrictHorizontalSizing, [961](#)
- setStrictVerticalSizing, [962](#)
- setStyle, [962](#)
- setStyleClass, [962](#)
- setStyleClassAssigned, [962](#)
- setVerticalStretchAffinity, [963](#)
- setVisibility, [963](#)
- setVstretch, [963](#)
- strictHorizontalSizing, [964](#)
- strictVerticalSizing, [964](#)
- style, [964](#)
- styleClass, [964](#)
- toggleSelectionBold, [964](#)
- toggleSelectionItalic, [964](#)
- toggleSelectionUnderline, [965](#)
- unregisterBusy, [965](#)
- verticalStretchAffinity, [965](#)
- visibility, [965](#)
- vstretch, [965](#)
- clove::RootNameScope, [966](#)
 - addChildNameScope, [966](#)
 - getByName, [967](#)
- clove::ScrollView, [967](#)
 - addStyleClass, [971](#)
 - bindProperty, [971](#)
 - body, [971](#)
 - bodySize, [971](#)
 - bodyWidget, [971](#)
 - busy, [972](#)
 - childrenWidgets, [972](#)
 - computeMinimalHeightForWidth, [972](#)
 - computeMinimalWidth, [972](#)
 - computePreferredHeightForWidth, [972](#)
 - computePreferredWidth, [973](#)
 - containingNameScope, [973](#)
 - declareProperty, [973](#)
 - doStandaloneResizing, [974](#)
 - doinit, [973](#)
 - doinitEarly, [974](#)
 - doresize, [974](#)
 - effectiveVisibility, [974](#)
 - effectivelyEnabled, [974](#)
 - enabled, [975](#)
 - focus, [975](#)
 - getMinimalHeightForWidth, [975](#)
 - getMinimalWidth, [975](#)
 - getPreferredHeightForWidth, [975](#)
 - getPreferredWidth, [976](#)
 - getProperty, [976](#)
 - hasFocus, [976](#)
 - horizontalScrollPosition, [976](#)
 - horizontalStretchAffinity, [977](#)
 - hstretch, [977](#)
 - init, [977](#)
 - innerSize, [977](#)
 - isAlive, [977](#)
 - isStyleClass, [978](#)
 - mayFocus, [978](#)
 - name, [978](#)
 - nameScope, [978](#)
 - OnDestroyed, [978](#)
 - OnKeyDown, [979](#)
 - OnKeyPress, [979](#)
 - OnKeyUp, [979](#)
 - OnPropertyChanged, [979](#)
 - OnResized, [979](#)
 - OnVisibilityChanged, [979](#)
 - outerSize, [980](#)
 - parentWidget, [980](#)
 - registerBusy, [980](#)
 - relayout, [980](#)
 - remove, [980](#)
 - removeStyleClass, [981](#)
 - resize, [981](#)
 - setBody, [981](#)
 - setBusy, [981](#)
 - setDoStandaloneResizing, [982](#)
 - setEnabled, [982](#)
 - setHorizontalScrollPosition, [982](#)
 - setHorizontalStretchAffinity, [982](#)
 - setHstretch, [983](#)
 - setMayFocus, [983](#)
 - setName, [983](#)
 - setProperty, [984](#)
 - setStrictHorizontalSizing, [984](#)
 - setStrictVerticalSizing, [984](#)
 - setStyle, [985](#)
 - setStyleClass, [985](#)
 - setStyleClassAssigned, [985](#)
 - setVerticalScrollPosition, [985](#)
 - setVerticalStretchAffinity, [987](#)
 - setVisibility, [987](#)
 - setVstretch, [987](#)
 - strictHorizontalSizing, [987](#)
 - strictVerticalSizing, [988](#)
 - style, [988](#)
 - styleClass, [988](#)
 - unregisterBusy, [988](#)
 - verticalScrollPosition, [988](#)
 - verticalStretchAffinity, [989](#)
 - visibility, [989](#)

- vstretch, 989
- clove::Slider, 989
 - addStyleClass, 993
 - bindProperty, 993
 - busy, 994
 - childrenWidgets, 994
 - computeMinimalHeightForWidth, 994
 - computeMinimalWidth, 994
 - computePreferredHeightForWidth, 994
 - computePreferredWidth, 995
 - containingNameScope, 995
 - declareProperty, 995
 - doStandaloneResizing, 996
 - doinit, 995
 - doinitEarly, 996
 - doresize, 996
 - effectiveVisibility, 996
 - effectivelyEnabled, 996
 - enabled, 997
 - focus, 997
 - getMinimalHeightForWidth, 997
 - getMinimalWidth, 997
 - getPreferredHeightForWidth, 997
 - getPreferredWidth, 998
 - getProperty, 998
 - hasFocus, 998
 - horizontalStretchAffinity, 998
 - hstretch, 999
 - init, 999
 - innerSize, 999
 - isAlive, 999
 - isStyleClass, 999
 - max, 1000
 - mayFocus, 1000
 - min, 1000
 - name, 1000
 - nameScope, 1000
 - OnChanged, 1001
 - OnDestroyed, 1001
 - OnKeyDown, 1001
 - OnKeyPress, 1001
 - OnKeyUp, 1001
 - OnPropertyChanged, 1001
 - OnResized, 1002
 - OnVisibilityChanged, 1002
 - orientation, 1002
 - outerSize, 1002
 - parentWidget, 1002
 - registerBusy, 1002
 - relayout, 1003
 - remove, 1003
 - removeStyleClass, 1003
 - resize, 1004
 - setBusy, 1004
 - setDoStandaloneResizing, 1004
 - setEnabled, 1004
 - setHorizontalStretchAffinity, 1005
 - setHstretch, 1005
 - setMax, 1005
 - setMayFocus, 1005
 - setMin, 1006
 - setName, 1006
 - setOrientation, 1006
 - setProperty, 1006
 - setStepsize, 1007
 - setStrictHorizontalSizing, 1007
 - setStrictVerticalSizing, 1007
 - setStyle, 1008
 - setStyleClass, 1008
 - setStyleClassAssigned, 1008
 - setValue, 1008
 - setVerticalStretchAffinity, 1009
 - setVisibility, 1009
 - setVstretch, 1009
 - stepsize, 1010
 - strictHorizontalSizing, 1010
 - strictVerticalSizing, 1010
 - style, 1010
 - styleClass, 1010
 - unregisterBusy, 1010
 - value, 1011
 - verticalStretchAffinity, 1011
 - visibility, 1011
 - vstretch, 1011
- clove::SortProxyDatasource, 1012
 - changeValue, 1013
 - columnCount, 1014
 - columnHeadersVisible, 1014
 - getColumnHeader, 1014
 - getMetadata, 1016
 - getRowHeader, 1016
 - getValue, 1016
 - nativePointerToProxyPointer, 1016
 - OnDataInsert, 1017
 - OnDataRemove, 1017
 - OnDataUpdate, 1017
 - OnHeaderDataInsert, 1017
 - OnHeaderDataRemove, 1017
 - OnHeaderDataUpdate, 1018
 - OnHeaderVisibilityUpdated, 1018
 - parent, 1018
 - proxyPointerToNativePointer, 1018
 - refresh, 1019
 - rowCount, 1019
 - rowHeadersVisible, 1019
 - setColumnComparator, 1019
 - setDatasource, 1020
 - setRowComparator, 1020
 - SortProxyDatasource, 1013
 - valuePointer, 1020
 - valuePointerNavigateInDepth, 1020
- clove::Spacer, 1021
 - addStyleClass, 1024
 - bindProperty, 1024
 - busy, 1025
 - childrenWidgets, 1025

- computeMinimalHeightForWidth, 1025
- computeMinimalWidth, 1025
- computePreferredHeightForWidth, 1026
- computePreferredWidth, 1026
- containingNameScope, 1026
- declareProperty, 1026
- doStandaloneResizing, 1027
- doinit, 1027
- doinitEarly, 1027
- doresize, 1027
- effectiveVisibility, 1028
- effectivelyEnabled, 1027
- enabled, 1028
- focus, 1028
- getMinimalHeightForWidth, 1028
- getMinimalWidth, 1028
- getPreferredHeightForWidth, 1029
- getPreferredWidth, 1029
- getProperty, 1029
- hasFocus, 1030
- horizontalStretchAffinity, 1030
- hstretch, 1030
- init, 1030
- innerSize, 1030
- isAlive, 1030
- isStyleClass, 1031
- mayFocus, 1031
- name, 1031
- nameScope, 1031
- OnDestroyed, 1031
- OnKeyDown, 1032
- OnKeyPress, 1032
- OnKeyUp, 1032
- OnPropertyChanged, 1032
- OnResized, 1032
- OnVisibilityChanged, 1032
- outerSize, 1033
- parentWidget, 1033
- registerBusy, 1033
- relayout, 1033
- remove, 1033
- removeStyleClass, 1034
- resize, 1034
- setBusy, 1034
- setDoStandaloneResizing, 1034
- setEnabled, 1035
- setHorizontalStretchAffinity, 1035
- setHstretch, 1035
- setMayFocus, 1035
- setName, 1036
- setProperty, 1036
- setStrictHorizontalSizing, 1036
- setStrictVerticalSizing, 1037
- setStyle, 1037
- setStyleClass, 1037
- setStyleClassAssigned, 1037
- setVerticalStretchAffinity, 1038
- setVisibility, 1038
- setVstretch, 1038
- strictHorizontalSizing, 1039
- strictVerticalSizing, 1039
- style, 1039
- styleClass, 1039
- unregisterBusy, 1039
- verticalStretchAffinity, 1040
- visibility, 1040
- vstretch, 1040
- clove::StackLayout, 1040
 - addChild, 1041
 - children, 1041
 - clearChilds, 1041
 - insertColumn, 1042
 - insertRow, 1042
 - removeColumn, 1042
 - removeRow, 1042
 - setChildren, 1043
- clove::TabView, 1075
 - addStyleClass, 1079
 - addTab, 1079
 - bindProperty, 1080
 - busy, 1080
 - childrenWidgets, 1080
 - computeMinimalHeightForWidth, 1080
 - computeMinimalWidth, 1081
 - computePreferredHeightForWidth, 1081
 - computePreferredWidth, 1081
 - containingNameScope, 1081
 - currentTab, 1081
 - declareProperty, 1082
 - doStandaloneResizing, 1083
 - doinit, 1082
 - doinitEarly, 1082
 - doresize, 1082
 - effectiveVisibility, 1083
 - effectivelyEnabled, 1083
 - enabled, 1083
 - focus, 1083
 - getMinimalHeightForWidth, 1083
 - getMinimalWidth, 1084
 - getPreferredHeightForWidth, 1084
 - getPreferredWidth, 1084
 - getProperty, 1084
 - hasFocus, 1085
 - horizontalStretchAffinity, 1085
 - hstretch, 1085
 - init, 1085
 - innerSize, 1085
 - isAlive, 1086
 - isStyleClass, 1086
 - mayFocus, 1086
 - name, 1086
 - nameScope, 1086
 - OnDestroyed, 1087
 - OnKeyDown, 1087
 - OnKeyPress, 1087
 - OnKeyUp, 1087

- OnPropertyChanged, 1087
- OnResized, 1087
- OnTabCreationRequested, 1088
- OnVisibilityChanged, 1088
- outerSize, 1088
- parentWidget, 1088
- registerBusy, 1088
- relayout, 1088
- remove, 1089
- removeStyleClass, 1089
- resize, 1089
- setBusy, 1089
- setCurrentTab, 1090
- setDoStandaloneResizing, 1090
- setEnabled, 1090
- setHorizontalStretchAffinity, 1091
- setHstretch, 1091
- setMayFocus, 1091
- setName, 1091
- setProperty, 1092
- setStrictHorizontalSizing, 1092
- setStrictVerticalSizing, 1092
- setStyle, 1093
- setStyleClass, 1093
- setStyleClassAssigned, 1093
- setSwitchInvisibleAnimationDuration, 1093
- setSwitchInvisibleAnimationName, 1094
- setSwitchVisibleAnimationDuration, 1094
- setSwitchVisibleAnimationName, 1094
- setTabBarLocation, 1095
- setTabs, 1095
- setUserMayAddTabs, 1095
- setVerticalStretchAffinity, 1095
- setVisibility, 1096
- setVstretch, 1096
- strictHorizontalSizing, 1096
- strictVerticalSizing, 1096
- style, 1097
- styleClass, 1097
- switchInvisibleAnimationDuration, 1097
- switchInvisibleAnimationName, 1097
- switchVisibleAnimationDuration, 1097
- switchVisibleAnimationName, 1097
- tabBarLocation, 1098
- tabs, 1098
- unregisterBusy, 1098
- userMayAddTabs, 1098
- verticalStretchAffinity, 1098
- visibility, 1099
- vstretch, 1099
- clove::TableView, 1044
 - addStyleClass, 1049
 - allowChecking, 1049
 - allowSelection, 1049
 - alwaysAllocateExpanderSpace, 1049
 - bindProperty, 1049
 - busy, 1050
 - cellGenerator, 1050
 - checkedCells, 1050
 - childrenWidgets, 1050
 - collapseCell, 1050
 - columnsResizable, 1051
 - computeMinimalHeightForWidth, 1051
 - computeMinimalWidth, 1051
 - computePreferredHeightForWidth, 1051
 - computePreferredWidth, 1052
 - containingNameScope, 1052
 - dataViewProcessor, 1052
 - datasource, 1052
 - declareProperty, 1052
 - doStandaloneResizing, 1053
 - doinit, 1053
 - doinitEarly, 1053
 - doresize, 1053
 - editCell, 1053
 - editOnGesture, 1054
 - effectiveVisibility, 1054
 - effectivelyEnabled, 1054
 - enabled, 1054
 - expandCell, 1054
 - expandCellRecursive, 1055
 - focus, 1055
 - getMinimalHeightForWidth, 1055
 - getMinimalWidth, 1055
 - getPreferredHeightForWidth, 1055
 - getPreferredWidth, 1056
 - getProperty, 1056
 - granularity, 1056
 - gridVisible, 1056
 - hasFocus, 1057
 - headersource, 1057
 - hideExpanders, 1057
 - horizontalStretchAffinity, 1057
 - hstretch, 1057
 - init, 1057
 - innerSize, 1058
 - isAlive, 1058
 - isCellChecked, 1058
 - isCellExpanded, 1058
 - isCellSelected, 1059
 - isStyleClass, 1059
 - mayFocus, 1059
 - name, 1059
 - nameScope, 1059
 - nodeActivationNeedsDoubleClick, 1059
 - OnDestroyed, 1060
 - OnKeyDown, 1060
 - OnKeyPress, 1060
 - OnKeyUp, 1060
 - OnPropertyChanged, 1060
 - OnResized, 1060
 - OnSelectionChanged, 1061
 - OnVisibilityChanged, 1061
 - outerSize, 1061
 - parentWidget, 1061
 - registerBusy, 1061

- relayout, [1061](#)
- remove, [1062](#)
- removeStyleClass, [1062](#)
- resize, [1062](#)
- rowsResizable, [1062](#)
- selectCell, [1063](#)
- selection, [1063](#)
- setAllowChecking, [1063](#)
- setAllowSelection, [1063](#)
- setAlwaysAllocateExpanderSpace, [1064](#)
- setBusy, [1064](#)
- setCellChecked, [1064](#)
- setCellGenerator, [1064](#)
- setCheckedCells, [1065](#)
- setColumnsResizable, [1065](#)
- setDataViewProcessor, [1066](#)
- setDatasource, [1065](#)
- setDoStandaloneResizing, [1066](#)
- setEditOnGesture, [1066](#)
- setEnabled, [1066](#)
- setGranularity, [1067](#)
- setGridVisible, [1067](#)
- setHeadersource, [1067](#)
- setHideExpanders, [1068](#)
- setHorizontalStretchAffinity, [1068](#)
- setHstretch, [1068](#)
- setMayFocus, [1068](#)
- setName, [1069](#)
- setNodeActivationNeedsDoubleClick, [1069](#)
- setProperty, [1069](#)
- setRowsResizable, [1070](#)
- setShowChangeMenu, [1070](#)
- setShowOnlyFirstColumn, [1070](#)
- setStrictHorizontalSizing, [1070](#)
- setStrictVerticalSizing, [1071](#)
- setStyle, [1071](#)
- setStyleClass, [1071](#)
- setStyleClassAssigned, [1072](#)
- setVerticalStretchAffinity, [1072](#)
- setVisibility, [1072](#)
- setVstretch, [1072](#)
- showChangeMenu, [1073](#)
- showOnlyFirstColumn, [1073](#)
- strictHorizontalSizing, [1073](#)
- strictVerticalSizing, [1073](#)
- style, [1074](#)
- styleClass, [1074](#)
- unregisterBusy, [1074](#)
- verticalStretchAffinity, [1074](#)
- visibility, [1074](#)
- vstretch, [1075](#)
- clove::TimeBox, [1099](#)
 - addStyleClass, [1103](#)
 - autocompleteFilter, [1104](#)
 - autocompleteItems, [1104](#)
 - autocompleteOpenForNoText, [1104](#)
 - bindProperty, [1104](#)
 - busy, [1104](#)
 - childrenWidgets, [1104](#)
 - computeMinimalHeightForWidth, [1105](#)
 - computeMinimalWidth, [1105](#)
 - computePreferredHeightForWidth, [1105](#)
 - computePreferredWidth, [1105](#)
 - containingNameScope, [1106](#)
 - declareProperty, [1106](#)
 - doStandaloneResizing, [1107](#)
 - doinit, [1106](#)
 - doinitEarly, [1106](#)
 - doresize, [1106](#)
 - effectiveVisibility, [1107](#)
 - effectivelyEnabled, [1107](#)
 - enabled, [1107](#)
 - focus, [1107](#)
 - getMinimalHeightForWidth, [1107](#)
 - getMinimalWidth, [1108](#)
 - getPreferredHeightForWidth, [1108](#)
 - getPreferredWidth, [1108](#)
 - getProperty, [1108](#)
 - hasFocus, [1109](#)
 - hintText, [1109](#)
 - horizontalStretchAffinity, [1109](#)
 - hstretch, [1109](#)
 - init, [1109](#)
 - innerSize, [1110](#)
 - isAlive, [1110](#)
 - isStyleClass, [1110](#)
 - mayFocus, [1110](#)
 - name, [1110](#)
 - nameScope, [1110](#)
 - OnChanged, [1111](#)
 - OnDestroyed, [1111](#)
 - OnKeyDown, [1111](#)
 - OnKeyPress, [1111](#)
 - OnKeyUp, [1111](#)
 - OnPopupTextSelected, [1111](#)
 - OnPropertyChanged, [1112](#)
 - OnResized, [1112](#)
 - OnVisibilityChanged, [1112](#)
 - outerSize, [1112](#)
 - parentWidget, [1112](#)
 - readOnly, [1112](#)
 - registerBusy, [1113](#)
 - relayout, [1113](#)
 - remove, [1113](#)
 - removeStyleClass, [1113](#)
 - resize, [1114](#)
 - setAutocompletionFilter, [1114](#)
 - setAutocompletionItems, [1114](#)
 - setAutocompletionOpenForNoText, [1114](#)
 - setBusy, [1115](#)
 - setDoStandaloneResizing, [1115](#)
 - setEnabled, [1115](#)
 - setHintText, [1115](#)
 - setHorizontalStretchAffinity, [1116](#)
 - setHstretch, [1116](#)
 - setMayFocus, [1116](#)

- setName, [1117](#)
- setProperty, [1117](#)
- setReadOnly, [1117](#)
- setStrictHorizontalSizing, [1118](#)
- setStrictVerticalSizing, [1118](#)
- setStyle, [1118](#)
- setStyleClass, [1118](#)
- setStyleClassAssigned, [1119](#)
- setText, [1119](#)
- setTextSelection, [1119](#)
- setTime, [1120](#)
- setVerticalStretchAffinity, [1120](#)
- setVisibility, [1120](#)
- setVstretch, [1120](#)
- strictHorizontalSizing, [1121](#)
- strictVerticalSizing, [1121](#)
- style, [1121](#)
- styleClass, [1121](#)
- text, [1121](#)
- textSelection, [1121](#)
- time, [1122](#)
- unregisterBusy, [1122](#)
- verticalStretchAffinity, [1122](#)
- visibility, [1122](#)
- vstretch, [1122](#)
- clove::ToolBar, [1123](#)
 - actions, [1126](#)
 - addStyleClass, [1127](#)
 - bindProperty, [1127](#)
 - busy, [1127](#)
 - childrenWidgets, [1127](#)
 - computeMinimalHeightForWidth, [1127](#)
 - computeMinimalWidth, [1128](#)
 - computePreferredHeightForWidth, [1128](#)
 - computePreferredWidth, [1128](#)
 - containingNameScope, [1128](#)
 - declareProperty, [1129](#)
 - doStandaloneResizing, [1130](#)
 - doinit, [1129](#)
 - doinitEarly, [1129](#)
 - doresize, [1129](#)
 - effectiveVisibility, [1130](#)
 - effectivelyEnabled, [1130](#)
 - enabled, [1130](#)
 - focus, [1130](#)
 - getMinimalHeightForWidth, [1130](#)
 - getMinimalWidth, [1131](#)
 - getPreferredHeightForWidth, [1131](#)
 - getPreferredWidth, [1131](#)
 - getProperty, [1131](#)
 - hasFocus, [1132](#)
 - head1, [1132](#)
 - head2, [1132](#)
 - headControl, [1132](#)
 - headControlWidget, [1132](#)
 - horizontalStretchAffinity, [1132](#)
 - hstretch, [1133](#)
 - icon, [1133](#)
 - init, [1133](#)
 - innerSize, [1133](#)
 - isAlive, [1133](#)
 - isStyleClass, [1134](#)
 - mayFocus, [1134](#)
 - name, [1134](#)
 - nameScope, [1134](#)
 - OnActionTriggered, [1134](#)
 - OnDestroyed, [1135](#)
 - OnKeyDown, [1135](#)
 - OnKeyPress, [1135](#)
 - OnKeyUp, [1135](#)
 - OnPropertyChanged, [1135](#)
 - OnResized, [1135](#)
 - OnVisibilityChanged, [1136](#)
 - outerSize, [1136](#)
 - parentWidget, [1136](#)
 - registerBusy, [1136](#)
 - relayout, [1136](#)
 - remove, [1136](#)
 - removeStyleClass, [1137](#)
 - resize, [1137](#)
 - setActions, [1137](#)
 - setBusy, [1137](#)
 - setDoStandaloneResizing, [1138](#)
 - setEnabled, [1138](#)
 - setHead1, [1138](#)
 - setHead2, [1138](#)
 - setHeadControl, [1139](#)
 - setHorizontalStretchAffinity, [1139](#)
 - setHstretch, [1139](#)
 - setIcon, [1140](#)
 - setMayFocus, [1140](#)
 - setName, [1140](#)
 - setProperty, [1140](#)
 - setStrictHorizontalSizing, [1141](#)
 - setStrictVerticalSizing, [1141](#)
 - setStyle, [1141](#)
 - setStyleClass, [1142](#)
 - setStyleClassAssigned, [1142](#)
 - setVerticalStretchAffinity, [1142](#)
 - setVisibility, [1142](#)
 - setVstretch, [1143](#)
 - strictHorizontalSizing, [1143](#)
 - strictVerticalSizing, [1143](#)
 - style, [1143](#)
 - styleClass, [1143](#)
 - unregisterBusy, [1144](#)
 - verticalStretchAffinity, [1144](#)
 - visibility, [1144](#)
 - vstretch, [1144](#)
- clove::TreeView, [1145](#)
 - addStyleClass, [1150](#)
 - allowChecking, [1150](#)
 - allowSelection, [1150](#)
 - alwaysAllocateExpanderSpace, [1150](#)
 - bindProperty, [1150](#)
 - busy, [1151](#)

- cellGenerator, 1151
- checkedCells, 1151
- childrenWidgets, 1151
- collapseCell, 1151
- columnsResizable, 1152
- computeMinimalHeightForWidth, 1152
- computeMinimalWidth, 1152
- computePreferredHeightForWidth, 1152
- computePreferredWidth, 1153
- containingNameScope, 1153
- dataViewProcessor, 1153
- datasource, 1153
- declareProperty, 1153
- doStandaloneResizing, 1154
- doinit, 1154
- doinitEarly, 1154
- doresize, 1154
- editCell, 1154
- editOnGesture, 1155
- effectiveVisibility, 1155
- effectivelyEnabled, 1155
- enabled, 1155
- expandCell, 1155
- expandCellRecursive, 1156
- focus, 1156
- getMinimalHeightForWidth, 1156
- getMinimalWidth, 1156
- getPreferredHeightForWidth, 1156
- getPreferredWidth, 1157
- getProperty, 1157
- granularity, 1157
- gridVisible, 1157
- hasFocus, 1158
- headersource, 1158
- hideExpanders, 1158
- horizontalStretchAffinity, 1158
- hstretch, 1158
- init, 1158
- innerSize, 1159
- isAlive, 1159
- isCellChecked, 1159
- isCellExpanded, 1159
- isCellSelected, 1160
- isStyleClass, 1160
- mayFocus, 1160
- name, 1160
- nameScope, 1160
- nodeActivationNeedsDoubleClick, 1160
- OnDestroyed, 1161
- OnKeyDown, 1161
- OnKeyPress, 1161
- OnKeyUp, 1161
- OnPropertyChanged, 1161
- OnResized, 1161
- OnSelectionChanged, 1162
- OnVisibilityChanged, 1162
- outerSize, 1162
- parentWidget, 1162
- registerBusy, 1162
- relayout, 1162
- remove, 1163
- removeStyleClass, 1163
- resize, 1163
- rowsResizable, 1163
- selectCell, 1164
- selection, 1164
- setAllowChecking, 1164
- setAllowSelection, 1164
- setAlwaysAllocateExpanderSpace, 1165
- setBusy, 1165
- setCellChecked, 1165
- setCellGenerator, 1165
- setCheckedCells, 1166
- setColumnsResizable, 1166
- setDataViewProcessor, 1167
- setDatasource, 1166
- setDoStandaloneResizing, 1167
- setEditOnGesture, 1167
- setEnabled, 1167
- setGranularity, 1168
- setGridVisible, 1168
- setHeadersource, 1168
- setHideExpanders, 1169
- setHorizontalStretchAffinity, 1169
- setHstretch, 1169
- setMayFocus, 1169
- setName, 1170
- setNodeActivationNeedsDoubleClick, 1170
- setProperty, 1170
- setRowsResizable, 1171
- setShowChangeMenu, 1171
- setShowOnlyFirstColumn, 1171
- setStrictHorizontalSizing, 1171
- setStrictVerticalSizing, 1172
- setStyle, 1172
- setStyleClass, 1172
- setStyleClassAssigned, 1173
- setVerticalStretchAffinity, 1173
- setVisibility, 1173
- setVstretch, 1173
- showChangeMenu, 1174
- showOnlyFirstColumn, 1174
- strictHorizontalSizing, 1174
- strictVerticalSizing, 1174
- style, 1175
- styleClass, 1175
- unregisterBusy, 1175
- verticalStretchAffinity, 1175
- visibility, 1175
- vstretch, 1176
- clove::VerticalStack, 1187
 - addChild, 1191
 - addStyleClass, 1191
 - bindProperty, 1191
 - busy, 1192
 - children, 1192

- childrenWidgets, 1192
- clearChilds, 1192
- computeMinimalHeightForWidth, 1192
- computeMinimalWidth, 1193
- computePreferredHeightForWidth, 1193
- computePreferredWidth, 1193
- containingNameScope, 1193
- declareProperty, 1193
- doStandaloneResizing, 1194
- doinit, 1194
- doinitEarly, 1194
- doresize, 1194
- effectiveVisibility, 1195
- effectivelyEnabled, 1195
- enabled, 1195
- focus, 1195
- getMinimalHeightForWidth, 1195
- getMinimalWidth, 1196
- getPreferredHeightForWidth, 1196
- getPreferredWidth, 1196
- getProperty, 1196
- hasFocus, 1197
- horizontalStretchAffinity, 1197
- hstretch, 1197
- init, 1197
- innerSize, 1197
- insertColumn, 1198
- insertRow, 1198
- isAlive, 1198
- isStyleClass, 1198
- mayFocus, 1199
- name, 1199
- nameScope, 1199
- OnDestroyed, 1199
- OnKeyDown, 1199
- OnKeyPress, 1199
- OnKeyUp, 1200
- OnPropertyChanged, 1200
- OnResized, 1200
- OnVisibilityChanged, 1200
- outerSize, 1200
- parentWidget, 1201
- registerBusy, 1201
- relayout, 1201
- remove, 1201
- removeColumn, 1202
- removeRow, 1202
- removeStyleClass, 1202
- resize, 1202
- rows, 1203
- setBusy, 1203
- setChildren, 1203
- setDoStandaloneResizing, 1203
- setEnabled, 1205
- setHorizontalStretchAffinity, 1205
- setHstretch, 1205
- setMayFocus, 1205
- setName, 1206
- setProperty, 1206
- setRows, 1206
- setStrictHorizontalSizing, 1207
- setStrictVerticalSizing, 1207
- setStyle, 1207
- setStyleClass, 1207
- setStyleClassAssigned, 1208
- setVerticalStretchAffinity, 1208
- setVisibility, 1208
- setVstretch, 1209
- strictHorizontalSizing, 1209
- strictVerticalSizing, 1209
- style, 1209
- styleClass, 1209
- unregisterBusy, 1210
- verticalStretchAffinity, 1210
- visibility, 1210
- vstretch, 1210
- clove::VideoPlayer, 1211
 - addStyleClass, 1215
 - autoPlay, 1215
 - bindProperty, 1215
 - busy, 1216
 - canPlayType, 1216
 - childrenWidgets, 1216
 - computeMinimalHeightForWidth, 1216
 - computeMinimalWidth, 1218
 - computePreferredHeightForWidth, 1218
 - computePreferredWidth, 1218
 - containingNameScope, 1218
 - currentMediaPosition, 1218
 - declareProperty, 1219
 - doStandaloneResizing, 1220
 - doinit, 1219
 - doinitEarly, 1219
 - doresize, 1219
 - duration, 1220
 - effectiveVisibility, 1220
 - effectivelyEnabled, 1220
 - enabled, 1220
 - focus, 1220
 - getMinimalHeightForWidth, 1221
 - getMinimalWidth, 1222
 - getPreferredHeightForWidth, 1222
 - getPreferredWidth, 1222
 - getProperty, 1222
 - hasEnded, 1223
 - hasFocus, 1223
 - horizontalStretchAffinity, 1223
 - hstretch, 1223
 - init, 1223
 - innerSize, 1224
 - isAlive, 1224
 - isPaused, 1224
 - isStyleClass, 1224
 - loop, 1224
 - mayFocus, 1225
 - muted, 1225

- name, 1225
- nameScope, 1225
- nativeNode, 1225
- OnDestroyed, 1225
- OnDurationChanged, 1225
- OnHasEnded, 1226
- OnIsPaused, 1226
- OnIsPlaying, 1226
- OnKeyDown, 1226
- OnKeyPress, 1226
- OnKeyUp, 1227
- OnLoadingAborted, 1227
- OnPropertyChanged, 1227
- OnReadyForPlayback, 1227
- OnResized, 1227
- OnVisibilityChanged, 1227
- outerSize, 1228
- parentWidget, 1228
- pause, 1228
- play, 1228
- preload, 1228
- registerBusy, 1228
- relayout, 1229
- remove, 1229
- removeStyleClass, 1229
- resize, 1230
- setAutoPlay, 1230
- setBusy, 1230
- setCurrentMediaPosition, 1230
- setDoStandaloneResizing, 1231
- setEnabled, 1231
- setHorizontalStretchAffinity, 1231
- setHstretch, 1231
- setLoop, 1232
- setMayFocus, 1232
- setMuted, 1232
- setName, 1232
- setPreload, 1233
- setProperty, 1233
- setShowControls, 1233
- setSource, 1234
- setStrictHorizontalSizing, 1234
- setStrictVerticalSizing, 1234
- setStyle, 1234
- setStyleClass, 1235
- setStyleClassAssigned, 1235
- setVerticalStretchAffinity, 1235
- setVisibility, 1236
- setVolume, 1236
- setVstretch, 1236
- showControls, 1236
- source, 1237
- strictHorizontalSizing, 1237
- strictVerticalSizing, 1237
- style, 1237
- styleClass, 1237
- unregisterBusy, 1237
- verticalStretchAffinity, 1238
- visibility, 1238
- volume, 1238
- vstretch, 1238
- close::Widget, 1239
 - addStyleClass, 1243
 - bindProperty, 1243
 - busy, 1243
 - childrenWidgets, 1243
 - computeMinimalHeightForWidth, 1244
 - computeMinimalWidth, 1244
 - computePreferredHeightForWidth, 1244
 - computePreferredWidth, 1244
 - containingNameScope, 1245
 - declareProperty, 1245
 - doStandaloneResizing, 1246
 - doinit, 1245
 - doinitEarly, 1245
 - doresize, 1245
 - effectiveVisibility, 1246
 - effectivelyEnabled, 1246
 - enabled, 1246
 - focus, 1246
 - getMinimalHeightForWidth, 1246
 - getMinimalWidth, 1247
 - getPreferredHeightForWidth, 1247
 - getPreferredWidth, 1247
 - getProperty, 1247
 - hasFocus, 1248
 - horizontalStretchAffinity, 1248
 - hstretch, 1248
 - init, 1248
 - innerSize, 1248
 - isAlive, 1249
 - isStyleClass, 1249
 - mayFocus, 1249
 - name, 1249
 - nameScope, 1249
 - OnDestroyed, 1250
 - OnKeyDown, 1250
 - OnKeyPress, 1250
 - OnKeyUp, 1250
 - OnPropertyChanged, 1250
 - OnResized, 1250
 - OnVisibilityChanged, 1251
 - outerSize, 1251
 - parentWidget, 1251
 - registerBusy, 1251
 - relayout, 1251
 - remove, 1251
 - removeStyleClass, 1252
 - resize, 1252
 - setBusy, 1252
 - setDoStandaloneResizing, 1252
 - setEnabled, 1253
 - setHorizontalStretchAffinity, 1253
 - setHstretch, 1253
 - setMayFocus, 1253
 - setName, 1254

- setProperty, [1254](#)
- setStrictHorizontalSizing, [1254](#)
- setStrictVerticalSizing, [1255](#)
- setStyle, [1255](#)
- setStyleClass, [1255](#)
- setStyleClassAssigned, [1255](#)
- setVerticalStretchAffinity, [1256](#)
- setVisibility, [1256](#)
- setVstretch, [1256](#)
- strictHorizontalSizing, [1257](#)
- strictVerticalSizing, [1257](#)
- style, [1257](#)
- styleClass, [1257](#)
- unregisterBusy, [1257](#)
- verticalStretchAffinity, [1258](#)
- visibility, [1258](#)
- vstretch, [1258](#)
- Widget, [1242](#)
- clove::Wrap, [1258](#)
 - addChild, [1262](#)
 - addStyleClass, [1262](#)
 - bindProperty, [1262](#)
 - busy, [1263](#)
 - children, [1263](#)
 - childrenWidgets, [1263](#)
 - clearChilds, [1263](#)
 - computeMinimalHeightForWidth, [1263](#)
 - computeMinimalWidth, [1264](#)
 - computePreferredHeightForWidth, [1264](#)
 - computePreferredWidth, [1264](#)
 - containingNameScope, [1264](#)
 - declareProperty, [1264](#)
 - doStandaloneResizing, [1265](#)
 - doinit, [1265](#)
 - doinitEarly, [1265](#)
 - doresize, [1265](#)
 - effectiveVisibility, [1266](#)
 - effectivelyEnabled, [1265](#)
 - enabled, [1266](#)
 - focus, [1266](#)
 - getMinimalHeightForWidth, [1266](#)
 - getMinimalWidth, [1266](#)
 - getPreferredHeightForWidth, [1267](#)
 - getPreferredWidth, [1267](#)
 - getProperty, [1267](#)
 - hasFocus, [1268](#)
 - horizontalStretchAffinity, [1268](#)
 - hstretch, [1268](#)
 - init, [1268](#)
 - innerSize, [1268](#)
 - isAlive, [1268](#)
 - isStyleClass, [1269](#)
 - mayFocus, [1269](#)
 - name, [1269](#)
 - nameScope, [1269](#)
 - OnDestroyed, [1269](#)
 - OnKeyDown, [1270](#)
 - OnKeyPress, [1270](#)
 - OnKeyUp, [1270](#)
 - OnPropertyChanged, [1270](#)
 - OnResized, [1270](#)
 - OnVisibilityChanged, [1270](#)
 - outerSize, [1271](#)
 - parentWidget, [1271](#)
 - registerBusy, [1271](#)
 - relayout, [1271](#)
 - remove, [1271](#)
 - removeStyleClass, [1272](#)
 - resize, [1272](#)
 - setBusy, [1272](#)
 - setChildren, [1272](#)
 - setDoStandaloneResizing, [1273](#)
 - setEnabled, [1273](#)
 - setHorizontalStretchAffinity, [1273](#)
 - setHstretch, [1273](#)
 - setMayFocus, [1274](#)
 - setName, [1274](#)
 - setProperty, [1274](#)
 - setStrictHorizontalSizing, [1275](#)
 - setStrictVerticalSizing, [1275](#)
 - setStyle, [1275](#)
 - setStyleClass, [1275](#)
 - setStyleClassAssigned, [1276](#)
 - setVerticalStretchAffinity, [1276](#)
 - setVisibility, [1276](#)
 - setVstretch, [1277](#)
 - strictHorizontalSizing, [1277](#)
 - strictVerticalSizing, [1277](#)
 - style, [1277](#)
 - styleClass, [1277](#)
 - unregisterBusy, [1278](#)
 - verticalStretchAffinity, [1278](#)
 - visibility, [1278](#)
 - vstretch, [1278](#)
- clove::WrapLayout, [1279](#)
 - addChild, [1279](#)
 - children, [1279](#)
 - clearChilds, [1280](#)
 - setChildren, [1280](#)
- clove::symbols, [1043](#)
- clove::utils, [1176](#)
 - addOnDestroyHandler, [1177](#)
 - addOnDoubleClickTapHandler, [1178](#)
 - addOnDragHandler, [1178](#)
 - animateNode, [1178](#)
 - applyDefaultsToConfig, [1179](#)
 - arrayToDatasource, [1179](#)
 - arraysum, [1179](#)
 - connectDatasourceToWidget, [1180](#)
 - cssLengthToPixels, [1181](#)
 - deferLoading, [1181](#)
 - findWidgetForDomNode, [1182](#)
 - getExtraSize, [1182](#)
 - getFullPathForLoadedResource, [1182](#)
 - getInnerSize, [1183](#)
 - getOuterSize, [1183](#)

- insertToArray, 1183
- isDescendantOf, 1184
- main_parts_resized, 1184
- OnMainResized, 1184
- popup, 1184
- removeOnDoubleClickTapHandler, 1185
- resumeResizing, 1185
- runOnLoaded, 1185
- setStyleClassAssigned, 1186
- suspendResizing, 1186
- tryLoadScript, 1186
- tryLoadStyle, 1187
- collapseCell
 - clove::DataView, 204
 - clove::ListView, 572
 - clove::TableView, 1050
 - clove::TreeView, 1151
- collapseHidden
 - clove::FlatLayout, 382
 - clove::FlatView, 391
- collapsedIcon
 - clove::Expander, 354
- collapsedLabel
 - clove::Expander, 354
- cols
 - clove::HorizontalStack, 461
- columnCount
 - clove::AjaxAsyncDatasource, 48
 - clove::AsyncDatasource, 58
 - clove::Datasource, 190
 - clove::DatasourceValuePointer, 194
 - clove::FilterProxyDatasource, 374
 - clove::NativeDatasource, 719
 - clove::NativeDatasourceNode, 729
 - clove::ProxyDatasource, 849
 - clove::SortProxyDatasource, 1014
- columnHeadersVisible
 - clove::AjaxAsyncDatasource, 48
 - clove::AsyncDatasource, 58
 - clove::FilterProxyDatasource, 374
 - clove::Headersource, 454
 - clove::NativeDatasource, 719
 - clove::ProxyDatasource, 850
 - clove::SortProxyDatasource, 1014
- columnsResizable
 - clove::DataView, 205
 - clove::ListView, 572
 - clove::TableView, 1051
 - clove::TreeView, 1152
- computeMinimalHeightForWidth
 - clove::AbstractMenu, 28
 - clove::AudioPlayer, 69
 - clove::Border, 94
 - clove::Button, 114
 - clove::Carousel, 136
 - clove::CheckButton, 161
 - clove::DataView, 205
 - clove::DateBox, 234
- clove::Dialog, 258
- clove::DropDownBox, 280
- clove::EditBox, 304
- clove::EditComboBox, 328
- clove::Expander, 354
- clove::FlatView, 391
- clove::Form, 413
- clove::Grid, 434
- clove::HorizontalStack, 461
- clove::HtmlView, 483
- clove::IconView, 506
- clove::ImageView, 526
- clove::Label, 547
- clove::ListView, 573
- clove::MainView, 603
- clove::MediaPlayer, 629
- clove::Menubar, 654
- clove::ModalPanel, 675
- clove::MultilineEditBox, 696
- clove::NumericEditBox, 739
- clove::PasswordEditBox, 765
- clove::PopupMenu, 787
- clove::PopupMenuButton, 809
- clove::ProgressBar, 831
- clove::RadioButton, 859
- clove::RawResizeSplitter, 884
- clove::ResizeSplitter, 905
- clove::RichEdit, 928
- clove::RichEditBox, 948
- clove::ScrollView, 972
- clove::Slider, 994
- clove::Spacer, 1025
- clove::TabView, 1080
- clove::TableView, 1051
- clove::TimeBox, 1105
- clove::ToolBar, 1127
- clove::TreeView, 1152
- clove::VerticalStack, 1192
- clove::VideoPlayer, 1216
- clove::Widget, 1244
- clove::Wrap, 1263
- computeMinimalWidth
 - clove::AbstractMenu, 29
 - clove::AudioPlayer, 69
 - clove::Border, 94
 - clove::Button, 115
 - clove::Carousel, 137
 - clove::CheckButton, 161
 - clove::DataView, 205
 - clove::DateBox, 236
 - clove::Dialog, 258
 - clove::DropDownBox, 280
 - clove::EditBox, 305
 - clove::EditComboBox, 328
 - clove::Expander, 355
 - clove::FlatView, 391
 - clove::Form, 413
 - clove::Grid, 434

- clove::HorizontalStack, 461
- clove::HtmlView, 483
- clove::IconView, 506
- clove::ImageView, 527
- clove::Label, 548
- clove::ListView, 573
- clove::MainView, 603
- clove::MediaPlayer, 629
- clove::Menubar, 655
- clove::ModalPanel, 675
- clove::MultilineEditBox, 697
- clove::NumericEditBox, 739
- clove::PasswordEditBox, 765
- clove::PopupMenu, 788
- clove::PopupMenuButton, 809
- clove::ProgressBar, 831
- clove::RadioButton, 860
- clove::RawResizeSplitter, 884
- clove::ResizeSplitter, 905
- clove::RichEdit, 929
- clove::RichEditBox, 949
- clove::ScrollView, 972
- clove::Slider, 994
- clove::Spacer, 1025
- clove::TabView, 1081
- clove::TableView, 1051
- clove::TimeBox, 1105
- clove::ToolBar, 1128
- clove::TreeView, 1152
- clove::VerticalStack, 1193
- clove::VideoPlayer, 1218
- clove::Widget, 1244
- clove::Wrap, 1264
- computePreferredHeightForWidth
 - clove::AbstractMenu, 29
 - clove::AudioPlayer, 69
 - clove::Border, 94
 - clove::Button, 115
 - clove::Carousel, 137
 - clove::CheckButton, 162
 - clove::DataView, 205
 - clove::DateBox, 236
 - clove::Dialog, 259
 - clove::DropDownBox, 281
 - clove::EditBox, 305
 - clove::EditComboBox, 328
 - clove::Expander, 355
 - clove::FlatView, 392
 - clove::Form, 414
 - clove::Grid, 434
 - clove::HorizontalStack, 462
 - clove::HtmlView, 484
 - clove::IconView, 507
 - clove::ImageView, 527
 - clove::Label, 548
 - clove::ListView, 573
 - clove::MainView, 604
 - clove::MediaPlayer, 630
 - clove::Menubar, 655
 - clove::ModalPanel, 675
 - clove::MultilineEditBox, 697
 - clove::NumericEditBox, 740
 - clove::PasswordEditBox, 765
 - clove::PopupMenu, 788
 - clove::PopupMenuButton, 810
 - clove::ProgressBar, 832
- computePreferredWidth
 - clove::AbstractMenu, 29
 - clove::AudioPlayer, 69
 - clove::Border, 95
 - clove::Button, 115
 - clove::Carousel, 137
 - clove::CheckButton, 162
 - clove::DataView, 206
 - clove::DateBox, 236
 - clove::Dialog, 259
 - clove::DropDownBox, 281
 - clove::EditBox, 305
 - clove::EditComboBox, 328
 - clove::Expander, 355
 - clove::FlatView, 392
 - clove::Form, 414
 - clove::Grid, 434
 - clove::HorizontalStack, 462
 - clove::HtmlView, 484
 - clove::IconView, 507
 - clove::ImageView, 527
 - clove::Label, 548
 - clove::ListView, 573
 - clove::MainView, 604
 - clove::MediaPlayer, 630
 - clove::Menubar, 655
 - clove::ModalPanel, 675
 - clove::MultilineEditBox, 697
 - clove::NumericEditBox, 740
 - clove::PasswordEditBox, 765
 - clove::PopupMenu, 788
 - clove::PopupMenuButton, 810
 - clove::ProgressBar, 832

- clove::RadioButton, 860
- clove::RawResizeSplitter, 885
- clove::ResizeSplitter, 906
- clove::RichEdit, 929
- clove::RichEditBox, 949
- clove::ScrollView, 973
- clove::Slider, 995
- clove::Spacer, 1026
- clove::TabView, 1081
- clove::TableView, 1052
- clove::TimeBox, 1105
- clove::Toolbar, 1128
- clove::TreeView, 1153
- clove::VerticalStack, 1193
- clove::VideoPlayer, 1218
- clove::Widget, 1244
- clove::Wrap, 1264
- connectDatasourceToWidget
 - clove::utils, 1180
- containingNameScope
 - clove::AbstractMenu, 29
 - clove::AudioPlayer, 70
 - clove::Border, 95
 - clove::Button, 115
 - clove::Carousel, 137
 - clove::CheckBox, 162
 - clove::DataView, 206
 - clove::DateBox, 236
 - clove::Dialog, 259
 - clove::DropDownBox, 281
 - clove::EditBox, 305
 - clove::EditComboBox, 329
 - clove::Expander, 355
 - clove::FlatView, 392
 - clove::Form, 414
 - clove::Grid, 435
 - clove::HorizontalStack, 462
 - clove::HtmlView, 484
 - clove::IconView, 507
 - clove::ImageView, 527
 - clove::Label, 548
 - clove::ListView, 574
 - clove::MainView, 604
 - clove::MediaPlayer, 630
 - clove::Menubar, 655
 - clove::ModalPanel, 676
 - clove::MultilineEditBox, 697
 - clove::NumericEditBox, 740
 - clove::PasswordEditBox, 766
 - clove::PopupMenu, 788
 - clove::PopupMenuButton, 810
 - clove::ProgressBar, 832
 - clove::RadioButton, 860
 - clove::RawResizeSplitter, 885
 - clove::ResizeSplitter, 906
 - clove::RichEdit, 929
 - clove::RichEditBox, 949
 - clove::ScrollView, 973
 - clove::Slider, 995
 - clove::Spacer, 1026
 - clove::TabView, 1081
 - clove::TableView, 1052
 - clove::TimeBox, 1106
 - clove::Toolbar, 1128
 - clove::TreeView, 1153
 - clove::VerticalStack, 1193
 - clove::VideoPlayer, 1218
 - clove::Widget, 1245
 - clove::Wrap, 1264
- contentHtmlNode
 - clove::RichEditBox, 949
- contentRoot
 - clove::HtmlView, 484
- conversationDialog
 - clove, 182
- convertFrom
 - clove::DataBindingConverter, 188
- convertTo
 - clove::DataBindingConverter, 189
- createHtml
 - clove::Icon, 501
- cssLengthToPixels
 - clove::utils, 1181
- currentMediaPosition
 - clove::AudioPlayer, 70
 - clove::MediaPlayer, 630
 - clove::VideoPlayer, 1218
- currentTab
 - clove::Carousel, 138
 - clove::TabView, 1081
- currentView
 - clove::FlatLayout, 382
 - clove::FlatView, 392
- DataBinding
 - clove::DataBinding, 187
- dataViewProcessor
 - clove::DataView, 206
 - clove::ListView, 574
 - clove::TableView, 1052
 - clove::TreeView, 1153
- databind
 - clove, 182
- datasource
 - clove::DataView, 206
 - clove::DatasourceValuePointer, 195
 - clove::ListView, 574
 - clove::TableView, 1052
 - clove::TreeView, 1153
- DatasourceValuePointer
 - clove::DatasourceValuePointer, 194
- date
 - clove::DateBox, 236
- declareProperty
 - clove::AbstractMenu, 29
 - clove::AudioPlayer, 70
 - clove::Border, 95

- clove::Button, 115
- clove::Carousel, 138
- clove::CheckBox, 162
- clove::DataView, 206
- clove::DateBox, 237
- clove::Dialog, 259
- clove::DropDownBox, 281
- clove::EditBox, 305
- clove::EditComboBox, 329
- clove::Expander, 355
- clove::FlatView, 392
- clove::Form, 414
- clove::Grid, 435
- clove::HorizontalStack, 462
- clove::HtmlView, 484
- clove::IconView, 507
- clove::ImageView, 527
- clove::Label, 548
- clove::ListView, 574
- clove::MainView, 604
- clove::MediaPlayer, 630
- clove::Menubar, 655
- clove::ModalPanel, 676
- clove::MultilineEditBox, 697
- clove::NumericEditBox, 740
- clove::PasswordEditBox, 766
- clove::PopupMenu, 788
- clove::PopupMenuButton, 810
- clove::ProgressBar, 832
- clove::RadioButton, 860
- clove::RawResizeSplitter, 885
- clove::ResizeSplitter, 906
- clove::RichEdit, 929
- clove::RichEditBox, 950
- clove::ScrollView, 973
- clove::Slider, 995
- clove::Spacer, 1026
- clove::TabView, 1082
- clove::TableView, 1052
- clove::TimeBox, 1106
- clove::Toolbar, 1129
- clove::TreeView, 1153
- clove::VerticalStack, 1193
- clove::VideoPlayer, 1219
- clove::Widget, 1245
- clove::Wrap, 1264
- deferLoading
 - clove::utils, 1181
- do_async_pull
 - clove::AjaxAsyncDatasource, 49
 - clove::AsyncDatasource, 58
- do_async_push
 - clove::AjaxAsyncDatasource, 49
 - clove::AsyncDatasource, 59
- doStandaloneResizing
 - clove::AbstractMenu, 30
 - clove::AudioPlayer, 71
 - clove::Border, 96
 - clove::Button, 116
 - clove::Carousel, 139
 - clove::CheckBox, 163
 - clove::DataView, 207
 - clove::DateBox, 238
 - clove::Dialog, 260
 - clove::DropDownBox, 282
 - clove::EditBox, 306
 - clove::EditComboBox, 330
 - clove::Expander, 356
 - clove::FlatView, 393
 - clove::Form, 415
 - clove::Grid, 436
 - clove::HorizontalStack, 463
 - clove::HtmlView, 485
 - clove::IconView, 508
 - clove::ImageView, 528
 - clove::Label, 549
 - clove::ListView, 575
 - clove::MainView, 605
 - clove::MediaPlayer, 631
 - clove::Menubar, 656
 - clove::ModalPanel, 677
 - clove::MultilineEditBox, 698
 - clove::NumericEditBox, 741
 - clove::PasswordEditBox, 767
 - clove::PopupMenu, 789
 - clove::PopupMenuButton, 811
 - clove::ProgressBar, 833
 - clove::RadioButton, 861
 - clove::RawResizeSplitter, 886
 - clove::ResizeSplitter, 907
 - clove::RichEdit, 930
 - clove::RichEditBox, 951
 - clove::ScrollView, 974
 - clove::Slider, 996
 - clove::Spacer, 1027
 - clove::TabView, 1083
 - clove::TableView, 1053
 - clove::TimeBox, 1107
 - clove::Toolbar, 1130
 - clove::TreeView, 1154
 - clove::VerticalStack, 1194
 - clove::VideoPlayer, 1220
 - clove::Widget, 1246
 - clove::Wrap, 1265
- doinit
 - clove::AbstractMenu, 30
 - clove::AudioPlayer, 70
 - clove::Border, 95
 - clove::Button, 116
 - clove::Carousel, 138
 - clove::CheckBox, 163
 - clove::DataView, 207
 - clove::DateBox, 237
 - clove::Dialog, 260
 - clove::DropDownBox, 282
 - clove::EditBox, 306

- clove::EditComboBox, 329
- clove::Expander, 356
- clove::FlatView, 392
- clove::Form, 415
- clove::Grid, 435
- clove::HorizontalStack, 462
- clove::HtmlView, 485
- clove::IconView, 507
- clove::ImageView, 528
- clove::Label, 549
- clove::ListView, 574
- clove::MainView, 604
- clove::MediaPlayer, 630
- clove::Menubar, 656
- clove::ModalPanel, 676
- clove::MultilineEditBox, 698
- clove::NumericEditBox, 740
- clove::PasswordEditBox, 766
- clove::PopupMenu, 789
- clove::PopupMenuButton, 810
- clove::ProgressBar, 832
- clove::RadioButton, 861
- clove::RawResizeSplitter, 885
- clove::ResizeSplitter, 906
- clove::RichEdit, 930
- clove::RichEditBox, 950
- clove::ScrollView, 973
- clove::Slider, 995
- clove::Spacer, 1027
- clove::TabView, 1082
- clove::TableView, 1053
- clove::TimeBox, 1106
- clove::ToolBar, 1129
- clove::TreeView, 1154
- clove::VerticalStack, 1194
- clove::VideoPlayer, 1219
- clove::Widget, 1245
- clove::Wrap, 1265
- doinitEarly
 - clove::AbstractMenu, 30
 - clove::AudioPlayer, 71
 - clove::Border, 96
 - clove::Button, 116
 - clove::Carousel, 138
 - clove::CheckButton, 163
 - clove::DataView, 207
 - clove::DateBox, 237
 - clove::Dialog, 260
 - clove::DropdownBox, 282
 - clove::EditBox, 306
 - clove::EditComboBox, 329
 - clove::Expander, 356
 - clove::FlatView, 393
 - clove::Form, 415
 - clove::Grid, 435
 - clove::HorizontalStack, 463
 - clove::HtmlView, 485
 - clove::IconView, 508
 - clove::ImageView, 528
 - clove::Label, 549
 - clove::ListView, 575
 - clove::MainView, 605
 - clove::MediaPlayer, 631
 - clove::Menubar, 656
 - clove::ModalPanel, 676
 - clove::MultilineEditBox, 698
- clove::ImageView, 528
- clove::Label, 549
- clove::ListView, 575
- clove::MainView, 605
- clove::MediaPlayer, 631
- clove::Menubar, 656
- clove::ModalPanel, 676
- clove::MultilineEditBox, 698
- clove::NumericEditBox, 741
- clove::PasswordEditBox, 766
- clove::PopupMenu, 789
- clove::PopupMenuButton, 810
- clove::ProgressBar, 832
- clove::RadioButton, 861
- clove::RawResizeSplitter, 886
- clove::ResizeSplitter, 906
- clove::RichEdit, 930
- clove::RichEditBox, 950
- clove::ScrollView, 974
- clove::Slider, 996
- clove::Spacer, 1027
- clove::TabView, 1082
- clove::TableView, 1053
- clove::TimeBox, 1106
- clove::ToolBar, 1129
- clove::TreeView, 1154
- clove::VerticalStack, 1194
- clove::VideoPlayer, 1219
- clove::Widget, 1245
- clove::Wrap, 1265
- doresize
 - clove::AbstractMenu, 30
 - clove::AudioPlayer, 71
 - clove::Border, 96
 - clove::Button, 116
 - clove::Carousel, 139
 - clove::CheckButton, 163
 - clove::DataView, 207
 - clove::DateBox, 237
 - clove::Dialog, 260
 - clove::DropdownBox, 282
 - clove::EditBox, 306
 - clove::EditComboBox, 329
 - clove::Expander, 356
 - clove::FlatView, 393
 - clove::Form, 415
 - clove::Grid, 435
 - clove::HorizontalStack, 463
 - clove::HtmlView, 485
 - clove::IconView, 508
 - clove::ImageView, 528
 - clove::Label, 549
 - clove::ListView, 575
 - clove::MainView, 605
 - clove::MediaPlayer, 631
 - clove::Menubar, 656
 - clove::ModalPanel, 676
 - clove::MultilineEditBox, 698

- clove::NumericEditBox, 741
- clove::PasswordEditBox, 766
- clove::PopupMenu, 789
- clove::PopupMenuButton, 811
- clove::ProgressBar, 833
- clove::RadioButton, 861
- clove::RawResizeSplitter, 886
- clove::ResizeSplitter, 907
- clove::RichEdit, 930
- clove::RichEditBox, 950
- clove::ScrollView, 974
- clove::Slider, 996
- clove::Spacer, 1027
- clove::TabView, 1082
- clove::TableView, 1053
- clove::TimeBox, 1106
- clove::Toolbar, 1129
- clove::TreeView, 1154
- clove::VerticalStack, 1194
- clove::VideoPlayer, 1219
- clove::Widget, 1245
- clove::Wrap, 1265
- duration
 - clove::AudioPlayer, 71
 - clove::MediaPlayer, 631
 - clove::VideoPlayer, 1220
- editCell
 - clove::DataView, 207
 - clove::ListView, 575
 - clove::TableView, 1053
 - clove::TreeView, 1154
- editOnGesture
 - clove::DataView, 208
 - clove::ListView, 576
 - clove::TableView, 1054
 - clove::TreeView, 1155
- effectiveVisibility
 - clove::AbstractMenu, 31
 - clove::AudioPlayer, 71
 - clove::Border, 96
 - clove::Button, 117
 - clove::Carousel, 139
 - clove::CheckButton, 164
 - clove::DataView, 208
 - clove::DateBox, 238
 - clove::Dialog, 261
 - clove::DropDownBox, 283
 - clove::EditBox, 307
 - clove::EditComboBox, 330
 - clove::Expander, 357
 - clove::FlatView, 393
 - clove::Form, 416
 - clove::Grid, 436
 - clove::HorizontalStack, 463
 - clove::HtmlView, 486
 - clove::IconView, 508
 - clove::ImageView, 529
 - clove::Label, 550
 - clove::ListView, 576
 - clove::MainView, 605
 - clove::MediaPlayer, 631
 - clove::Menubar, 657
 - clove::ModalPanel, 677
 - clove::MultilineEditBox, 699
 - clove::NumericEditBox, 741
 - clove::PasswordEditBox, 767
 - clove::PopupMenu, 790
 - clove::PopupMenuButton, 811
 - clove::ProgressBar, 833
 - clove::RadioButton, 862
 - clove::RawResizeSplitter, 886
 - clove::ResizeSplitter, 907
 - clove::RichEdit, 931
 - clove::RichEditBox, 951
 - clove::ScrollView, 974
 - clove::Slider, 996
 - clove::Spacer, 1028
 - clove::TabView, 1083
 - clove::TableView, 1054
 - clove::TimeBox, 1107
 - clove::Toolbar, 1130
 - clove::TreeView, 1155
 - clove::VerticalStack, 1195
 - clove::VideoPlayer, 1220
 - clove::Widget, 1246
 - clove::Wrap, 1266
- effectivelyEnabled
 - clove::AbstractMenu, 31
 - clove::AudioPlayer, 71
 - clove::Border, 96
 - clove::Button, 117
 - clove::Carousel, 139
 - clove::CheckButton, 163
 - clove::DataView, 208
 - clove::DateBox, 238
 - clove::Dialog, 260
 - clove::DropDownBox, 282
 - clove::EditBox, 307
 - clove::EditComboBox, 330
 - clove::Expander, 357
 - clove::FlatView, 393
 - clove::Form, 415
 - clove::Grid, 436
 - clove::HorizontalStack, 463
 - clove::HtmlView, 485
 - clove::IconView, 508
 - clove::ImageView, 529
 - clove::Label, 550
 - clove::ListView, 576
 - clove::MainView, 605
 - clove::MediaPlayer, 631
 - clove::Menubar, 657
 - clove::ModalPanel, 677
 - clove::MultilineEditBox, 699
 - clove::NumericEditBox, 741
 - clove::PasswordEditBox, 767

- clove::PopupMenu, 790
- clove::PopupMenuButton, 811
- clove::ProgressBar, 833
- clove::RadioButton, 862
- clove::RawResizeSplitter, 886
- clove::ResizeSplitter, 907
- clove::RichEdit, 931
- clove::RichEditBox, 951
- clove::ScrollView, 974
- clove::Slider, 996
- clove::Spacer, 1027
- clove::TabView, 1083
- clove::TableView, 1054
- clove::TimeBox, 1107
- clove::ToolBar, 1130
- clove::TreeView, 1155
- clove::VerticalStack, 1195
- clove::VideoPlayer, 1220
- clove::Widget, 1246
- clove::Wrap, 1265
- enabled
 - clove::AbstractMenu, 31
 - clove::AudioPlayer, 72
 - clove::Border, 97
 - clove::Button, 117
 - clove::Carousel, 139
 - clove::CheckBox, 164
 - clove::DataView, 208
 - clove::DateBox, 238
 - clove::Dialog, 261
 - clove::DropDownBox, 283
 - clove::EditBox, 307
 - clove::EditComboBox, 330
 - clove::Expander, 357
 - clove::FlatView, 394
 - clove::Form, 416
 - clove::Grid, 436
 - clove::HorizontalStack, 464
 - clove::HtmlView, 486
 - clove::IconView, 508
 - clove::ImageView, 529
 - clove::Label, 550
 - clove::ListView, 576
 - clove::MainView, 606
 - clove::MediaPlayer, 632
 - clove::Menubar, 657
 - clove::ModalPanel, 677
 - clove::MultilineEditBox, 699
 - clove::NumericEditBox, 742
 - clove::PasswordEditBox, 767
 - clove::PopupMenu, 790
 - clove::PopupMenuButton, 811
 - clove::ProgressBar, 833
 - clove::RadioButton, 862
 - clove::RawResizeSplitter, 887
 - clove::ResizeSplitter, 907
 - clove::RichEdit, 931
 - clove::RichEditBox, 951
 - clove::ScrollView, 975
 - clove::Slider, 997
 - clove::Spacer, 1028
 - clove::TabView, 1083
 - clove::TableView, 1054
 - clove::TimeBox, 1107
 - clove::ToolBar, 1130
 - clove::TreeView, 1155
 - clove::VerticalStack, 1195
 - clove::VideoPlayer, 1220
 - clove::Widget, 1246
 - clove::Wrap, 1266
- equals
 - clove::DatasourceValuePointer, 195
- Event
 - clove::Event, 346
- EventArgs
 - clove::EventArgs, 349
- expandCell
 - clove::DataView, 208
 - clove::ListView, 576
 - clove::TableView, 1054
 - clove::TreeView, 1155
- expandCellRecursive
 - clove::DataView, 209
 - clove::ListView, 577
 - clove::TableView, 1055
 - clove::TreeView, 1156
- expandedIcon
 - clove::Expander, 357
- expandedLabel
 - clove::Expander, 357
- expanderIndicatorDirection
 - clove::PopupMenu, 790
- FilterProxyDatasource
 - clove::FilterProxyDatasource, 372
- findWidgetForDomNode
 - clove::utils, 1182
- focus
 - clove::AbstractMenu, 31
 - clove::AudioPlayer, 72
 - clove::Border, 97
 - clove::Button, 117
 - clove::Carousel, 140
 - clove::CheckBox, 164
 - clove::DataView, 209
 - clove::DateBox, 238
 - clove::Dialog, 261
 - clove::DropDownBox, 283
 - clove::EditBox, 307
 - clove::EditComboBox, 330
 - clove::Expander, 357
 - clove::FlatView, 394
 - clove::Form, 416
 - clove::Grid, 436
 - clove::HorizontalStack, 464
 - clove::HtmlView, 486
 - clove::IconView, 509

- clove::ImageView, [529](#)
- clove::Label, [550](#)
- clove::ListView, [577](#)
- clove::MainView, [606](#)
- clove::MediaPlayer, [632](#)
- clove::Menubar, [657](#)
- clove::ModalPanel, [677](#)
- clove::MultilineEditBox, [699](#)
- clove::NumericEditBox, [742](#)
- clove::PasswordEditBox, [767](#)
- clove::PopupMenu, [790](#)
- clove::PopupMenuButton, [812](#)
- clove::ProgressBar, [834](#)
- clove::RadioButton, [862](#)
- clove::RawResizeSplitter, [887](#)
- clove::ResizeSplitter, [908](#)
- clove::RichEdit, [931](#)
- clove::RichEditBox, [951](#)
- clove::ScrollView, [975](#)
- clove::Slider, [997](#)
- clove::Spacer, [1028](#)
- clove::TabView, [1083](#)
- clove::TableView, [1055](#)
- clove::TimeBox, [1107](#)
- clove::ToolBar, [1130](#)
- clove::TreeView, [1156](#)
- clove::VerticalStack, [1195](#)
- clove::VideoPlayer, [1220](#)
- clove::Widget, [1246](#)
- clove::Wrap, [1266](#)
- focusBuddy
 - clove::Label, [550](#)
- fromPath
 - clove::DatasourceValuePointer, [195](#)
- fullyTransparent
 - clove::ModalPanel, [677](#)
- getAjaxId
 - clove::AjaxAsyncDatasource, [49](#)
- getByName
 - clove, [183](#)
 - clove::NameScope, [715](#)
 - clove::RootNameScope, [967](#)
- getChild
 - clove::NativeDatasourceNode, [729](#)
- getColumnHeader
 - clove::AjaxAsyncDatasource, [50](#)
 - clove::AsyncDatasource, [59](#)
 - clove::FilterProxyDatasource, [375](#)
 - clove::Headersource, [454](#)
 - clove::NativeDatasource, [719](#)
 - clove::ProxyDatasource, [850](#)
 - clove::SortProxyDatasource, [1014](#)
- getExtraSize
 - clove::utils, [1182](#)
- getFullPathForLoadedResource
 - clove::utils, [1182](#)
- getGroupByPublicName
 - clove::RadioGroup, [878](#)
- getInnerSize
 - clove::utils, [1183](#)
- getMetadata
 - clove::AjaxAsyncDatasource, [50](#)
 - clove::AsyncDatasource, [59](#)
 - clove::Datasource, [191](#)
 - clove::FilterProxyDatasource, [375](#)
 - clove::NativeDatasource, [719](#)
 - clove::NativeDatasourceNode, [729](#)
 - clove::ProxyDatasource, [850](#)
 - clove::SortProxyDatasource, [1016](#)
- getMinimalHeightForWidth
 - clove::AbstractMenu, [31](#)
 - clove::AudioPlayer, [72](#)
 - clove::Border, [97](#)
 - clove::Button, [117](#)
 - clove::Carousel, [140](#)
 - clove::CheckButton, [164](#)
 - clove::DataView, [209](#)
 - clove::DateBox, [238](#)
 - clove::Dialog, [261](#)
 - clove::DropDownBox, [283](#)
 - clove::EditBox, [307](#)
 - clove::EditComboBox, [330](#)
 - clove::Expander, [357](#)
 - clove::FlatView, [394](#)
 - clove::Form, [416](#)
 - clove::Grid, [436](#)
 - clove::HorizontalStack, [464](#)
 - clove::HtmlView, [486](#)
 - clove::IconView, [509](#)
 - clove::ImageView, [529](#)
 - clove::Label, [550](#)
 - clove::ListView, [577](#)
 - clove::MainView, [606](#)
 - clove::MediaPlayer, [632](#)
 - clove::Menubar, [657](#)
 - clove::ModalPanel, [678](#)
 - clove::MultilineEditBox, [699](#)
 - clove::NumericEditBox, [742](#)
 - clove::PasswordEditBox, [767](#)
 - clove::PopupMenu, [790](#)
 - clove::PopupMenuButton, [812](#)
 - clove::ProgressBar, [834](#)
 - clove::RadioButton, [862](#)
 - clove::RawResizeSplitter, [887](#)
 - clove::ResizeSplitter, [908](#)
 - clove::RichEdit, [931](#)
 - clove::RichEditBox, [951](#)
 - clove::ScrollView, [975](#)
 - clove::Slider, [997](#)
 - clove::Spacer, [1028](#)
 - clove::TabView, [1083](#)
 - clove::TableView, [1055](#)
 - clove::TimeBox, [1107](#)
 - clove::ToolBar, [1130](#)
 - clove::TreeView, [1156](#)
 - clove::VerticalStack, [1195](#)

- clove::VideoPlayer, 1221
- clove::Widget, 1246
- clove::Wrap, 1266
- getMinimalWidth
 - clove::AbstractMenu, 32
 - clove::AudioPlayer, 72
 - clove::Border, 97
 - clove::Button, 118
 - clove::Carousel, 140
 - clove::CheckBox, 164
 - clove::DataView, 209
 - clove::DateBox, 239
 - clove::Dialog, 261
 - clove::DropDownBox, 283
 - clove::EditBox, 308
 - clove::EditComboBox, 331
 - clove::Expander, 358
 - clove::FlatView, 394
 - clove::Form, 416
 - clove::Grid, 437
 - clove::HorizontalStack, 464
 - clove::HtmlView, 486
 - clove::IconView, 509
 - clove::ImageView, 530
 - clove::Label, 551
 - clove::ListView, 577
 - clove::MainView, 606
 - clove::MediaPlayer, 632
 - clove::Menubar, 658
 - clove::ModalPanel, 679
 - clove::MultilineEditBox, 700
 - clove::NumericEditBox, 742
 - clove::PasswordEditBox, 768
 - clove::PopupMenu, 791
 - clove::PopupMenuButton, 812
 - clove::ProgressBar, 834
 - clove::RadioButton, 863
 - clove::RawResizeSplitter, 887
 - clove::ResizeSplitter, 908
 - clove::RichEdit, 932
 - clove::RichEditBox, 952
 - clove::ScrollView, 975
 - clove::Slider, 997
 - clove::Spacer, 1028
 - clove::TabView, 1084
 - clove::TableView, 1055
 - clove::TimeBox, 1108
 - clove::Toolbar, 1131
 - clove::TreeView, 1156
 - clove::VerticalStack, 1196
 - clove::VideoPlayer, 1222
 - clove::Widget, 1247
 - clove::Wrap, 1266
- getOuterSize
 - clove::utils, 1183
- getPreferredHeightForWidth
 - clove::AbstractMenu, 32
 - clove::AudioPlayer, 72
 - clove::Border, 97
 - clove::Button, 118
 - clove::Carousel, 141
 - clove::CheckBox, 165
 - clove::DataView, 210
 - clove::DateBox, 239
 - clove::Dialog, 262
 - clove::DropDownBox, 284
- clove::Border, 97
- clove::Button, 118
- clove::Carousel, 140
- clove::CheckBox, 165
- clove::DataView, 209
- clove::DateBox, 239
- clove::Dialog, 262
- clove::DropDownBox, 284
- clove::EditBox, 308
- clove::EditComboBox, 331
- clove::Expander, 358
- clove::FlatView, 394
- clove::Form, 417
- clove::Grid, 437
- clove::HorizontalStack, 464
- clove::HtmlView, 487
- clove::IconView, 509
- clove::ImageView, 530
- clove::Label, 551
- clove::ListView, 577
- clove::MainView, 606
- clove::MediaPlayer, 633
- clove::Menubar, 658
- clove::ModalPanel, 679
- clove::MultilineEditBox, 700
- clove::NumericEditBox, 742
- clove::PasswordEditBox, 768
- clove::PopupMenu, 791
- clove::PopupMenuButton, 812
- clove::ProgressBar, 834
- clove::RadioButton, 863
- clove::RawResizeSplitter, 887
- clove::ResizeSplitter, 908
- clove::RichEdit, 932
- clove::RichEditBox, 952
- clove::ScrollView, 975
- clove::Slider, 997
- clove::Spacer, 1029
- clove::TabView, 1084
- clove::TableView, 1055
- clove::TimeBox, 1108
- clove::Toolbar, 1131
- clove::TreeView, 1156
- clove::VerticalStack, 1196
- clove::VideoPlayer, 1222
- clove::Widget, 1247
- clove::Wrap, 1267
- getPreferredWidth
 - clove::AbstractMenu, 32
 - clove::AudioPlayer, 73
 - clove::Border, 98
 - clove::Button, 118
 - clove::Carousel, 141
 - clove::CheckBox, 165
 - clove::DataView, 210
 - clove::DateBox, 239
 - clove::Dialog, 262
 - clove::DropDownBox, 284

- clove::EditBox, 308
- clove::EditComboBox, 331
- clove::Expander, 358
- clove::FlatView, 395
- clove::Form, 417
- clove::Grid, 437
- clove::HorizontalStack, 465
- clove::HtmlView, 487
- clove::IconView, 510
- clove::ImageView, 530
- clove::Label, 551
- clove::ListView, 578
- clove::MainView, 607
- clove::MediaPlayer, 633
- clove::Menubar, 658
- clove::ModalPanel, 679
- clove::MultilineEditBox, 700
- clove::NumericEditBox, 743
- clove::PasswordEditBox, 768
- clove::PopupMenu, 791
- clove::PopupMenuButton, 813
- clove::ProgressBar, 835
- clove::RadioButton, 863
- clove::RawResizeSplitter, 888
- clove::ResizeSplitter, 909
- clove::RichEdit, 932
- clove::RichEditBox, 952
- clove::ScrollView, 976
- clove::Slider, 998
- clove::Spacer, 1029
- clove::TabView, 1084
- clove::TableView, 1056
- clove::TimeBox, 1108
- clove::Toolbar, 1131
- clove::TreeView, 1157
- clove::VerticalStack, 1196
- clove::VideoPlayer, 1222
- clove::Widget, 1247
- clove::Wrap, 1267
- getProperty
 - clove::AbstractMenu, 32
 - clove::AudioPlayer, 73
 - clove::Border, 98
 - clove::Button, 118
 - clove::Carousel, 141
 - clove::CheckButton, 165
 - clove::DataView, 210
 - clove::DateBox, 239
 - clove::Dialog, 262
 - clove::DropDownBox, 284
 - clove::EditBox, 308
 - clove::EditComboBox, 331
 - clove::Expander, 358
 - clove::FlatView, 395
 - clove::Form, 417
 - clove::Grid, 437
 - clove::HorizontalStack, 465
 - clove::HtmlView, 487
 - clove::IconView, 510
 - clove::ImageView, 530
 - clove::Label, 551
 - clove::ListView, 578
 - clove::MainView, 607
 - clove::MediaPlayer, 633
 - clove::Menubar, 658
 - clove::ModalPanel, 679
 - clove::MultilineEditBox, 700
 - clove::NumericEditBox, 743
 - clove::PasswordEditBox, 768
 - clove::PopupMenu, 791
 - clove::PopupMenuButton, 813
 - clove::ProgressBar, 835
 - clove::RadioButton, 863
 - clove::RawResizeSplitter, 888
 - clove::ResizeSplitter, 909
 - clove::RichEdit, 932
 - clove::RichEditBox, 952
 - clove::ScrollView, 976
 - clove::Slider, 998
 - clove::Spacer, 1029
 - clove::TabView, 1084
 - clove::TableView, 1056
 - clove::TimeBox, 1108
 - clove::Toolbar, 1131
 - clove::TreeView, 1157
 - clove::VerticalStack, 1196
 - clove::VideoPlayer, 1222
 - clove::Widget, 1247
 - clove::Wrap, 1267
- getRowHeader
 - clove::AjaxAsyncDatasource, 50
 - clove::AsyncDatasource, 60
 - clove::FilterProxyDatasource, 375
 - clove::Headersource, 455
 - clove::NativeDatasource, 720
 - clove::ProxyDatasource, 850
 - clove::SortProxyDatasource, 1016
- getValue
 - clove::AjaxAsyncDatasource, 51
 - clove::AsyncDatasource, 60
 - clove::Datasource, 191
 - clove::DatasourceValuePointer, 195
 - clove::FilterProxyDatasource, 376
 - clove::NativeDatasource, 720
 - clove::NativeDatasourceNode, 729
 - clove::ProxyDatasource, 851
 - clove::SortProxyDatasource, 1016
- granularity
 - clove::DataView, 210
 - clove::ListView, 578
 - clove::TableView, 1056
 - clove::TreeView, 1157
- gridVisible
 - clove::DataView, 210
 - clove::ListView, 578
 - clove::TableView, 1056

- clove::TreeView, 1157
- group
 - clove::RadioButton, 864
- groupValue
 - clove::RadioButton, 864
- hasEnded
 - clove::AudioPlayer, 73
 - clove::MediaPlayer, 634
 - clove::VideoPlayer, 1223
- hasFocus
 - clove::AbstractMenu, 33
 - clove::AudioPlayer, 73
 - clove::Border, 98
 - clove::Button, 119
 - clove::Carousel, 141
 - clove::CheckBox, 166
 - clove::DataView, 211
 - clove::DateBox, 240
 - clove::Dialog, 263
 - clove::DropDownBox, 285
 - clove::EditBox, 309
 - clove::EditComboBox, 332
 - clove::Expander, 359
 - clove::FlatView, 395
 - clove::Form, 418
 - clove::Grid, 438
 - clove::HorizontalStack, 465
 - clove::HtmlView, 488
 - clove::IconView, 510
 - clove::ImageView, 531
 - clove::Label, 552
 - clove::ListView, 579
 - clove::MainView, 607
 - clove::MediaPlayer, 634
 - clove::Menubar, 659
 - clove::ModalPanel, 680
 - clove::MultilineEditBox, 701
 - clove::NumericEditBox, 743
 - clove::PasswordEditBox, 769
 - clove::PopupMenu, 792
 - clove::PopupMenuButton, 813
 - clove::ProgressBar, 835
 - clove::RadioButton, 864
 - clove::RawResizeSplitter, 888
 - clove::ResizeSplitter, 909
 - clove::RichEdit, 933
 - clove::RichEditBox, 953
 - clove::ScrollView, 976
 - clove::Slider, 998
 - clove::Spacer, 1030
 - clove::TabView, 1085
 - clove::TableView, 1057
 - clove::TimeBox, 1109
 - clove::Toolbar, 1132
 - clove::TreeView, 1158
 - clove::VerticalStack, 1197
 - clove::VideoPlayer, 1223
 - clove::Widget, 1248
 - clove::Wrap, 1268
- hasHandlers
 - clove::Event, 347
- head1
 - clove::MainView, 607
 - clove::Toolbar, 1132
- head2
 - clove::MainView, 608
 - clove::Toolbar, 1132
- headControl
 - clove::MainView, 608
 - clove::Toolbar, 1132
- headControlWidget
 - clove::MainView, 608
 - clove::Toolbar, 1132
- headersource
 - clove::DataView, 211
 - clove::ListView, 579
 - clove::TableView, 1057
 - clove::TreeView, 1158
- hideExpanders
 - clove::DataView, 211
 - clove::ListView, 579
 - clove::TableView, 1057
 - clove::TreeView, 1158
- hintText
 - clove::DateBox, 240
 - clove::DropDownBox, 285
 - clove::EditBox, 309
 - clove::EditComboBox, 332
 - clove::MultilineEditBox, 701
 - clove::NumericEditBox, 743
 - clove::PasswordEditBox, 769
 - clove::TimeBox, 1109
- horizontalScrollPosition
 - clove::ScrollView, 976
- horizontalStretchAffinity
 - clove::AbstractMenu, 33
 - clove::AudioPlayer, 74
 - clove::Border, 98
 - clove::Button, 119
 - clove::Carousel, 141
 - clove::CheckBox, 166
 - clove::DataView, 211
 - clove::DateBox, 240
 - clove::Dialog, 263
 - clove::DropDownBox, 285
 - clove::EditBox, 309
 - clove::EditComboBox, 332
 - clove::Expander, 359
 - clove::FlatView, 395
 - clove::Form, 418
 - clove::Grid, 438
 - clove::HorizontalStack, 465
 - clove::HtmlView, 488
 - clove::IconView, 510
 - clove::ImageView, 531
 - clove::Label, 552

- clove::ListView, 579
- clove::MainView, 608
- clove::MediaPlayer, 634
- clove::Menubar, 659
- clove::ModalPanel, 680
- clove::MultilineEditBox, 701
- clove::NumericEditBox, 744
- clove::PasswordEditBox, 769
- clove::PopupMenu, 792
- clove::PopupMenuButton, 813
- clove::ProgressBar, 835
- clove::RadioButton, 864
- clove::RawResizeSplitter, 888
- clove::ResizeSplitter, 909
- clove::RichEdit, 933
- clove::RichEditBox, 953
- clove::ScrollView, 977
- clove::Slider, 998
- clove::Spacer, 1030
- clove::TabView, 1085
- clove::TableView, 1057
- clove::TimeBox, 1109
- clove::ToolBar, 1132
- clove::TreeView, 1158
- clove::VerticalStack, 1197
- clove::VideoPlayer, 1223
- clove::Widget, 1248
- clove::Wrap, 1268
- hstretch
 - clove::AbstractMenu, 33
 - clove::AudioPlayer, 74
 - clove::Border, 99
 - clove::Button, 119
 - clove::Carousel, 141
 - clove::CheckBox, 166
 - clove::DataView, 211
 - clove::DateBox, 240
 - clove::Dialog, 263
 - clove::DropDownBox, 285
 - clove::EditBox, 309
 - clove::EditComboBox, 332
 - clove::Expander, 359
 - clove::FlatView, 396
 - clove::Form, 418
 - clove::Grid, 438
 - clove::HorizontalStack, 466
 - clove::HtmlView, 488
 - clove::IconView, 510
 - clove::ImageView, 531
 - clove::Label, 552
 - clove::ListView, 579
 - clove::MainView, 608
 - clove::MediaPlayer, 634
 - clove::Menubar, 659
 - clove::ModalPanel, 680
 - clove::MultilineEditBox, 701
 - clove::NumericEditBox, 744
 - clove::PasswordEditBox, 769
 - clove::PopupMenu, 792
 - clove::PopupMenuButton, 813
 - clove::ProgressBar, 835
 - clove::RadioButton, 864
 - clove::RawResizeSplitter, 888
 - clove::ResizeSplitter, 909
 - clove::RichEdit, 933
 - clove::RichEditBox, 953
 - clove::ScrollView, 977
 - clove::Slider, 998
 - clove::Spacer, 1030
 - clove::TabView, 1085
 - clove::TableView, 1057
 - clove::TimeBox, 1109
 - clove::ToolBar, 1133
 - clove::TreeView, 1158
 - clove::VerticalStack, 1197
 - clove::VideoPlayer, 1223
 - clove::Widget, 1248
 - clove::Wrap, 1268
- htmlContent
 - clove::Label, 552
 - clove::RichEdit, 933
 - clove::RichEditBox, 953
- i18n
 - clove, 183
- icol
 - clove::DatasourceValuePointer, 196
- icon
 - clove::Button, 119
 - clove::IconView, 511
 - clove::MainView, 608
 - clove::PopupMenuButton, 814
 - clove::ToolBar, 1133
- init
 - clove::AbstractMenu, 33
 - clove::AudioPlayer, 74
 - clove::Border, 99
 - clove::Button, 119
 - clove::Carousel, 142
 - clove::CheckBox, 166
 - clove::DataView, 211
 - clove::DateBox, 240
 - clove::Dialog, 263
 - clove::DropDownBox, 285
 - clove::EditBox, 309
 - clove::EditComboBox, 332
 - clove::Expander, 359
 - clove::FlatView, 396
 - clove::Form, 418
 - clove::Grid, 438
 - clove::HorizontalStack, 466
 - clove::HtmlView, 488
 - clove::IconView, 511
 - clove::ImageView, 531
 - clove::Label, 552
 - clove::ListView, 579
 - clove::MainView, 609

- clove::MediaPlayer, 634
- clove::Menubar, 659
- clove::ModalPanel, 680
- clove::MultilineEditBox, 701
- clove::NumericEditBox, 744
- clove::PasswordEditBox, 769
- clove::PopupMenu, 792
- clove::PopupMenuButton, 814
- clove::ProgressBar, 836
- clove::RadioButton, 864
- clove::RawResizeSplitter, 889
- clove::ResizeSplitter, 910
- clove::RichEdit, 933
- clove::RichEditBox, 953
- clove::ScrollView, 977
- clove::Slider, 999
- clove::Spacer, 1030
- clove::TabView, 1085
- clove::TableView, 1057
- clove::TimeBox, 1109
- clove::Toolbar, 1133
- clove::TreeView, 1158
- clove::VerticalStack, 1197
- clove::VideoPlayer, 1223
- clove::Widget, 1248
- clove::Wrap, 1268
- innerSize
 - clove::AbstractMenu, 33
 - clove::AudioPlayer, 74
 - clove::Border, 99
 - clove::Button, 120
 - clove::Carousel, 142
 - clove::CheckBox, 166
 - clove::DataView, 212
 - clove::DateBox, 241
 - clove::Dialog, 263
 - clove::DropDownBox, 286
 - clove::EditBox, 310
 - clove::EditComboBox, 333
 - clove::Expander, 359
 - clove::FlatView, 396
 - clove::Form, 418
 - clove::Grid, 438
 - clove::HorizontalStack, 466
 - clove::HtmlView, 488
 - clove::IconView, 511
 - clove::ImageView, 531
 - clove::Label, 553
 - clove::ListView, 580
 - clove::MainView, 609
 - clove::MediaPlayer, 635
 - clove::Menubar, 659
 - clove::ModalPanel, 680
 - clove::MultilineEditBox, 702
 - clove::NumericEditBox, 744
 - clove::PasswordEditBox, 770
 - clove::PopupMenu, 792
 - clove::PopupMenuButton, 814
 - clove::ProgressBar, 836
 - clove::RadioButton, 865
 - clove::RawResizeSplitter, 889
 - clove::ResizeSplitter, 910
 - clove::RichEdit, 934
 - clove::RichEditBox, 954
 - clove::ScrollView, 977
 - clove::Slider, 999
 - clove::Spacer, 1030
 - clove::TabView, 1085
 - clove::TableView, 1058
 - clove::TimeBox, 1110
 - clove::Toolbar, 1133
 - clove::TreeView, 1159
 - clove::VerticalStack, 1197
 - clove::VideoPlayer, 1224
 - clove::Widget, 1248
 - clove::Wrap, 1268
- inputDialog
 - clove, 183
- insertBodyWidget
 - clove::ResizeSplitter, 910
- insertColumn
 - clove::Grid, 439
 - clove::GridLayout, 452
 - clove::HorizontalStack, 466
 - clove::NativeDatasource, 720
 - clove::NativeDatasourceNode, 730
 - clove::StackLayout, 1042
 - clove::VerticalStack, 1198
- insertRow
 - clove::Grid, 439
 - clove::GridLayout, 452
 - clove::HorizontalStack, 467
 - clove::NativeDatasource, 721
 - clove::NativeDatasourceNode, 730
 - clove::StackLayout, 1042
 - clove::VerticalStack, 1198
- insertToArray
 - clove::utils, 1183
- interval
 - clove::Carousel, 142
- Invisible
 - clove, 184
- InvisibleCollapsed
 - clove, 184
- irow
 - clove::DatasourceValuePointer, 196
- isAlive
 - clove::AbstractMenu, 34
 - clove::AudioPlayer, 74
 - clove::Border, 99
 - clove::Button, 120
 - clove::Carousel, 142
 - clove::CheckBox, 166
 - clove::DataView, 212
 - clove::DateBox, 241
 - clove::Dialog, 263

- clove::DropDownBox, [286](#)
- clove::EditBox, [310](#)
- clove::EditComboBox, [333](#)
- clove::Expander, [360](#)
- clove::FlatView, [396](#)
- clove::Form, [418](#)
- clove::Grid, [439](#)
- clove::HorizontalStack, [467](#)
- clove::HtmlView, [488](#)
- clove::IconView, [511](#)
- clove::ImageView, [532](#)
- clove::Label, [553](#)
- clove::ListView, [580](#)
- clove::MainView, [609](#)
- clove::MediaPlayer, [635](#)
- clove::Menubar, [660](#)
- clove::ModalPanel, [681](#)
- clove::MultilineEditBox, [702](#)
- clove::NumericEditBox, [744](#)
- clove::PasswordEditBox, [770](#)
- clove::PopupMenu, [793](#)
- clove::PopupMenuButton, [814](#)
- clove::ProgressBar, [836](#)
- clove::RadioButton, [865](#)
- clove::RawResizeSplitter, [889](#)
- clove::ResizeSplitter, [910](#)
- clove::RichEdit, [934](#)
- clove::RichEditBox, [954](#)
- clove::ScrollView, [977](#)
- clove::Slider, [999](#)
- clove::Spacer, [1030](#)
- clove::TabView, [1086](#)
- clove::TableView, [1058](#)
- clove::TimeBox, [1110](#)
- clove::ToolBar, [1133](#)
- clove::TreeView, [1159](#)
- clove::VerticalStack, [1198](#)
- clove::VideoPlayer, [1224](#)
- clove::Widget, [1249](#)
- clove::Wrap, [1268](#)
- isCellChecked
 - clove::DataView, [212](#)
 - clove::ListView, [580](#)
 - clove::TableView, [1058](#)
 - clove::TreeView, [1159](#)
- isCellExpanded
 - clove::DataView, [212](#)
 - clove::ListView, [580](#)
 - clove::TableView, [1058](#)
 - clove::TreeView, [1159](#)
- isCellSelected
 - clove::DataView, [213](#)
 - clove::ListView, [581](#)
 - clove::TableView, [1059](#)
 - clove::TreeView, [1160](#)
- isDescendantOf
 - clove::utils, [1184](#)
- isExpanded
 - clove::Expander, [360](#)
- isPaused
 - clove::AudioPlayer, [75](#)
 - clove::MediaPlayer, [635](#)
 - clove::VideoPlayer, [1224](#)
- isPlaying
 - clove::Carousel, [142](#)
- isStyleClass
 - clove::AbstractMenu, [34](#)
 - clove::AudioPlayer, [75](#)
 - clove::Border, [99](#)
 - clove::Button, [120](#)
 - clove::Carousel, [143](#)
 - clove::CheckButton, [167](#)
 - clove::DataView, [213](#)
 - clove::DateBox, [241](#)
 - clove::Dialog, [264](#)
 - clove::DropDownBox, [286](#)
 - clove::EditBox, [310](#)
 - clove::EditComboBox, [333](#)
 - clove::Expander, [360](#)
 - clove::FlatView, [396](#)
 - clove::Form, [419](#)
 - clove::Grid, [439](#)
 - clove::HorizontalStack, [467](#)
 - clove::HtmlView, [489](#)
 - clove::IconView, [511](#)
 - clove::ImageView, [532](#)
 - clove::Label, [553](#)
 - clove::ListView, [581](#)
 - clove::MainView, [609](#)
 - clove::MediaPlayer, [635](#)
 - clove::Menubar, [660](#)
 - clove::ModalPanel, [681](#)
 - clove::MultilineEditBox, [702](#)
 - clove::NumericEditBox, [745](#)
 - clove::PasswordEditBox, [770](#)
 - clove::PopupMenu, [793](#)
 - clove::PopupMenuButton, [814](#)
 - clove::ProgressBar, [836](#)
 - clove::RadioButton, [865](#)
 - clove::RawResizeSplitter, [889](#)
 - clove::ResizeSplitter, [911](#)
 - clove::RichEdit, [934](#)
 - clove::RichEditBox, [954](#)
 - clove::ScrollView, [978](#)
 - clove::Slider, [999](#)
 - clove::Spacer, [1031](#)
 - clove::TabView, [1086](#)
 - clove::TableView, [1059](#)
 - clove::TimeBox, [1110](#)
 - clove::ToolBar, [1134](#)
 - clove::TreeView, [1160](#)
 - clove::VerticalStack, [1198](#)
 - clove::VideoPlayer, [1224](#)
 - clove::Widget, [1249](#)
 - clove::Wrap, [1269](#)
- keepAspectRatio

- clove::ImageView, 532
- label
 - clove::Button, 120
 - clove::CheckBox, 167
 - clove::Label, 553
 - clove::PopupMenuButton, 815
 - clove::ProgressBar, 837
 - clove::RadioButton, 865
- labelFunction
 - clove::ProgressBar, 837
- loop
 - clove::AudioPlayer, 75
 - clove::MediaPlayer, 635
 - clove::VideoPlayer, 1224
- main_parts_resized
 - clove::utils, 1184
- max
 - clove::NumericEditBox, 745
 - clove::Slider, 1000
- mayFocus
 - clove::AbstractMenu, 34
 - clove::AudioPlayer, 75
 - clove::Border, 100
 - clove::Button, 120
 - clove::Carousel, 143
 - clove::CheckBox, 167
 - clove::DataView, 213
 - clove::DateBox, 241
 - clove::Dialog, 264
 - clove::DropDownBox, 286
 - clove::EditBox, 310
 - clove::EditComboBox, 333
 - clove::Expander, 360
 - clove::FlatView, 397
 - clove::Form, 419
 - clove::Grid, 440
 - clove::HorizontalStack, 467
 - clove::HtmlView, 489
 - clove::IconView, 512
 - clove::ImageView, 532
 - clove::Label, 553
 - clove::ListView, 581
 - clove::MainView, 610
 - clove::MediaPlayer, 635
 - clove::Menubar, 660
 - clove::ModalPanel, 681
 - clove::MultilineEditBox, 702
 - clove::NumericEditBox, 745
 - clove::PasswordEditBox, 770
 - clove::PopupMenu, 793
 - clove::PopupMenuButton, 815
 - clove::ProgressBar, 837
 - clove::RadioButton, 865
 - clove::RawResizeSplitter, 890
 - clove::ResizeSplitter, 911
 - clove::RichEdit, 934
 - clove::RichEditBox, 954
 - clove::ScrollView, 978
 - clove::Slider, 1000
 - clove::Spacer, 1031
 - clove::TabView, 1086
 - clove::TableView, 1059
 - clove::TimeBox, 1110
 - clove::ToolBar, 1134
 - clove::TreeView, 1160
 - clove::VerticalStack, 1199
 - clove::VideoPlayer, 1225
 - clove::Widget, 1249
 - clove::Wrap, 1269
- MenuSeparator
 - clove, 184
- messageDialog
 - clove, 184
- min
 - clove::NumericEditBox, 745
 - clove::Slider, 1000
- muted
 - clove::AudioPlayer, 75
 - clove::MediaPlayer, 636
 - clove::VideoPlayer, 1225
- name
 - clove::AbstractMenu, 34
 - clove::AudioPlayer, 76
 - clove::Border, 100
 - clove::Button, 120
 - clove::Carousel, 143
 - clove::CheckBox, 167
 - clove::DataView, 213
 - clove::DateBox, 241
 - clove::Dialog, 264
 - clove::DropDownBox, 286
 - clove::EditBox, 310
 - clove::EditComboBox, 333
 - clove::Expander, 360
 - clove::FlatView, 397
 - clove::Form, 419
 - clove::Grid, 440
 - clove::HorizontalStack, 467
 - clove::HtmlView, 489
 - clove::IconView, 512
 - clove::ImageView, 532
 - clove::Label, 553
 - clove::ListView, 581
 - clove::MainView, 610
 - clove::MediaPlayer, 636
 - clove::Menubar, 660
 - clove::ModalPanel, 681
 - clove::MultilineEditBox, 702
 - clove::NumericEditBox, 745
 - clove::PasswordEditBox, 770
 - clove::PopupMenu, 793
 - clove::PopupMenuButton, 815
 - clove::ProgressBar, 837
 - clove::RadioButton, 865
 - clove::RawResizeSplitter, 890

- clove::ResizeSplitter, 911
- clove::RichEdit, 934
- clove::RichEditBox, 954
- clove::ScrollView, 978
- clove::Slider, 1000
- clove::Spacer, 1031
- clove::TabView, 1086
- clove::TableView, 1059
- clove::TimeBox, 1110
- clove::ToolBar, 1134
- clove::TreeView, 1160
- clove::VerticalStack, 1199
- clove::VideoPlayer, 1225
- clove::Widget, 1249
- clove::Wrap, 1269
- nameScope
 - clove::AbstractMenu, 34
 - clove::AudioPlayer, 76
 - clove::Border, 100
 - clove::Button, 121
 - clove::Carousel, 143
 - clove::CheckBox, 167
 - clove::DataView, 213
 - clove::DateBox, 241
 - clove::Dialog, 264
 - clove::DropDownBox, 286
 - clove::EditBox, 310
 - clove::EditComboBox, 333
 - clove::Expander, 360
 - clove::FlatView, 397
 - clove::Form, 419
 - clove::Grid, 440
 - clove::HorizontalStack, 468
 - clove::HtmlView, 489
 - clove::IconView, 512
 - clove::ImageView, 533
 - clove::Label, 554
 - clove::ListView, 581
 - clove::MainView, 610
 - clove::MediaPlayer, 636
 - clove::Menubar, 660
 - clove::ModalPanel, 681
 - clove::MultilineEditBox, 702
 - clove::NumericEditBox, 746
 - clove::PasswordEditBox, 770
 - clove::PopupMenu, 793
 - clove::PopupMenuButton, 815
 - clove::ProgressBar, 837
 - clove::RadioButton, 866
 - clove::RawResizeSplitter, 890
 - clove::ResizeSplitter, 911
 - clove::RichEdit, 934
 - clove::RichEditBox, 954
 - clove::ScrollView, 978
 - clove::Slider, 1000
 - clove::Spacer, 1031
 - clove::TabView, 1086
 - clove::TableView, 1059
 - clove::TimeBox, 1110
 - clove::ToolBar, 1134
 - clove::TreeView, 1160
 - clove::VerticalStack, 1199
 - clove::VideoPlayer, 1225
 - clove::Widget, 1249
 - clove::Wrap, 1269
- NativeDatasource
 - clove::NativeDatasource, 717
- nativeNode
 - clove::AudioPlayer, 76
 - clove::MediaPlayer, 636
 - clove::VideoPlayer, 1225
- nativePointerToProxyPointer
 - clove::FilterProxyDatasource, 376
 - clove::ProxyDatasource, 851
 - clove::SortProxyDatasource, 1016
- next
 - clove::Carousel, 143
- nodeActivationNeedsDoubleClick
 - clove::DataView, 213
 - clove::ListView, 581
 - clove::TableView, 1059
 - clove::TreeView, 1160
- notifications
 - clove, 185
- notify
 - clove::NotificationController, 733
- OnActionTriggered
 - clove::AbstractMenu, 35
 - clove::Menubar, 661
 - clove::PopupMenu, 794
 - clove::PopupMenuButton, 815
 - clove::ToolBar, 1134
- OnBeforeSubactionsExpanded
 - clove::AbstractMenu, 35
 - clove::Menubar, 661
 - clove::PopupMenu, 794
- OnChanged
 - clove::CheckBox, 168
 - clove::DateBox, 242
 - clove::DropDownBox, 287
 - clove::EditBox, 311
 - clove::EditComboBox, 334
 - clove::MultilineEditBox, 703
 - clove::NumericEditBox, 746
 - clove::PasswordEditBox, 771
 - clove::RadioButton, 866
 - clove::RadioGroup, 878
 - clove::Slider, 1001
 - clove::TimeBox, 1111
- OnClicked
 - clove::Button, 121
 - clove::CheckBox, 168
 - clove::ModalPanel, 682
 - clove::PopupMenuButton, 816
 - clove::RadioButton, 866
- OnDataArrived

- clove::AjaxAsyncDatasource, 51
- OnDataInsert
 - clove::AjaxAsyncDatasource, 51
 - clove::AsyncDatasource, 60
 - clove::Datasource, 191
 - clove::FilterProxyDatasource, 376
 - clove::NativeDatasource, 721
 - clove::ProxyDatasource, 851
 - clove::SortProxyDatasource, 1017
- OnDataRemove
 - clove::AjaxAsyncDatasource, 51
 - clove::AsyncDatasource, 60
 - clove::Datasource, 191
 - clove::FilterProxyDatasource, 376
 - clove::NativeDatasource, 721
 - clove::ProxyDatasource, 851
 - clove::SortProxyDatasource, 1017
- OnDataUpdate
 - clove::AjaxAsyncDatasource, 51
 - clove::AsyncDatasource, 60
 - clove::Datasource, 191
 - clove::FilterProxyDatasource, 376
 - clove::NativeDatasource, 721
 - clove::ProxyDatasource, 852
 - clove::SortProxyDatasource, 1017
- OnDestroyed
 - clove::AbstractMenu, 35
 - clove::AudioPlayer, 76
 - clove::Border, 100
 - clove::Button, 121
 - clove::Carousel, 144
 - clove::CheckBox, 168
 - clove::DataView, 214
 - clove::DateBox, 242
 - clove::Dialog, 264
 - clove::DropDownBox, 287
 - clove::EditBox, 311
 - clove::EditComboBox, 334
 - clove::Expander, 361
 - clove::FlatView, 397
 - clove::Form, 419
 - clove::Grid, 440
 - clove::HorizontalStack, 468
 - clove::HtmlView, 489
 - clove::IconView, 512
 - clove::ImageView, 533
 - clove::Label, 554
 - clove::ListView, 582
 - clove::MainView, 610
 - clove::MediaPlayer, 636
 - clove::MenuBar, 661
 - clove::ModalPanel, 682
 - clove::MultilineEditBox, 703
 - clove::NumericEditBox, 746
 - clove::PasswordEditBox, 771
 - clove::PopupMenu, 794
 - clove::PopupMenuButton, 816
 - clove::ProgressBar, 837
 - clove::RadioButton, 866
 - clove::RawResizeSplitter, 890
 - clove::ResizeSplitter, 911
 - clove::RichEdit, 935
 - clove::RichEditBox, 955
 - clove::ScrollView, 978
 - clove::Slider, 1001
 - clove::Spacer, 1031
 - clove::TabView, 1087
 - clove::TableView, 1060
 - clove::TimeBox, 1111
 - clove::ToolBar, 1135
 - clove::TreeView, 1161
 - clove::VerticalStack, 1199
 - clove::VideoPlayer, 1225
 - clove::Widget, 1250
 - clove::Wrap, 1269
- OnDurationChanged
 - clove::AudioPlayer, 76
 - clove::MediaPlayer, 636
 - clove::VideoPlayer, 1225
- OnHasEnded
 - clove::AudioPlayer, 76
 - clove::MediaPlayer, 637
 - clove::VideoPlayer, 1226
- OnHeaderDataInsert
 - clove::AjaxAsyncDatasource, 51, 52
 - clove::AsyncDatasource, 61
 - clove::FilterProxyDatasource, 377
 - clove::Headersource, 455
 - clove::NativeDatasource, 721
 - clove::ProxyDatasource, 852
 - clove::SortProxyDatasource, 1017
- OnHeaderDataRemove
 - clove::AjaxAsyncDatasource, 52
 - clove::AsyncDatasource, 61
 - clove::FilterProxyDatasource, 377
 - clove::Headersource, 455
 - clove::NativeDatasource, 722
 - clove::ProxyDatasource, 852
 - clove::SortProxyDatasource, 1017
- OnHeaderDataUpdate
 - clove::AjaxAsyncDatasource, 52
 - clove::AsyncDatasource, 61
 - clove::FilterProxyDatasource, 377
 - clove::Headersource, 455
 - clove::NativeDatasource, 722
 - clove::ProxyDatasource, 852
 - clove::SortProxyDatasource, 1018
- OnHeaderVisibilityUpdated
 - clove::AjaxAsyncDatasource, 52, 53
 - clove::AsyncDatasource, 61
 - clove::FilterProxyDatasource, 377
 - clove::Headersource, 455
 - clove::NativeDatasource, 722
 - clove::ProxyDatasource, 852
 - clove::SortProxyDatasource, 1018
- OnIsPaused

- clove::AudioPlayer, 77
- clove::MediaPlayer, 637
- clove::VideoPlayer, 1226
- OnIsPlaying
 - clove::AudioPlayer, 77
 - clove::MediaPlayer, 637
 - clove::VideoPlayer, 1226
- OnKeyDown
 - clove::AbstractMenu, 35
 - clove::AudioPlayer, 77
 - clove::Border, 100
 - clove::Button, 121
 - clove::Carousel, 144
 - clove::CheckBox, 168
 - clove::DataView, 214
 - clove::DateBox, 242
 - clove::Dialog, 265
 - clove::DropDownBox, 287
 - clove::EditBox, 311
 - clove::EditComboBox, 334
 - clove::Expander, 361
 - clove::FlatView, 398
 - clove::Form, 420
 - clove::Grid, 440
 - clove::HorizontalStack, 468
 - clove::HtmlView, 490
 - clove::IconView, 512
 - clove::ImageView, 533
 - clove::Label, 554
 - clove::ListView, 582
 - clove::MainView, 610
 - clove::MediaPlayer, 637
 - clove::Menubar, 661
 - clove::ModalPanel, 682
 - clove::MultilineEditBox, 703
 - clove::NumericEditBox, 746
 - clove::PasswordEditBox, 771
 - clove::PopupMenu, 794
 - clove::PopupMenuButton, 816
 - clove::ProgressBar, 838
 - clove::RadioButton, 866
 - clove::RawResizeSplitter, 890
 - clove::ResizeSplitter, 912
 - clove::RichEdit, 935
 - clove::RichEditBox, 955
 - clove::ScrollView, 979
 - clove::Slider, 1001
 - clove::Spacer, 1032
 - clove::TabView, 1087
 - clove::TableView, 1060
 - clove::TimeBox, 1111
 - clove::ToolBar, 1135
 - clove::TreeView, 1161
 - clove::VerticalStack, 1199
 - clove::VideoPlayer, 1226
 - clove::Widget, 1250
 - clove::Wrap, 1270
- OnKeyPress
 - clove::AbstractMenu, 35
 - clove::AudioPlayer, 77
 - clove::Border, 101
 - clove::Button, 121
 - clove::Carousel, 144
 - clove::CheckBox, 168
 - clove::DataView, 214
 - clove::DateBox, 242
 - clove::AbstractMenu, 35
 - clove::AudioPlayer, 77
 - clove::Border, 101
 - clove::Button, 121
 - clove::Carousel, 144
 - clove::CheckBox, 168
 - clove::DataView, 214
 - clove::DateBox, 242
 - clove::Dialog, 265
 - clove::DropDownBox, 287
 - clove::EditBox, 311
 - clove::EditComboBox, 334
 - clove::Expander, 361
 - clove::FlatView, 398
 - clove::Form, 420
 - clove::Grid, 440
 - clove::HorizontalStack, 468
 - clove::HtmlView, 490
 - clove::IconView, 513
 - clove::ImageView, 533
 - clove::Label, 554
 - clove::ListView, 582
 - clove::MainView, 610
 - clove::MediaPlayer, 637
 - clove::Menubar, 661
 - clove::ModalPanel, 682
 - clove::MultilineEditBox, 703
 - clove::NumericEditBox, 746
 - clove::PasswordEditBox, 771
 - clove::PopupMenu, 794
 - clove::PopupMenuButton, 816
 - clove::ProgressBar, 838
 - clove::RadioButton, 866
 - clove::RawResizeSplitter, 891
 - clove::ResizeSplitter, 912
 - clove::RichEdit, 935
 - clove::RichEditBox, 955
 - clove::ScrollView, 979
 - clove::Slider, 1001
 - clove::Spacer, 1032
 - clove::TabView, 1087
 - clove::TableView, 1060
 - clove::TimeBox, 1111
 - clove::ToolBar, 1135
 - clove::TreeView, 1161
 - clove::VerticalStack, 1199
 - clove::VideoPlayer, 1226
 - clove::Widget, 1250
 - clove::Wrap, 1270
- OnKeyUp
 - clove::AbstractMenu, 35
 - clove::AudioPlayer, 77
 - clove::Border, 101
 - clove::Button, 121
 - clove::Carousel, 144
 - clove::CheckBox, 168
 - clove::DataView, 214
 - clove::DateBox, 242

- clove::Dialog, [265](#)
- clove::DropDownBox, [287](#)
- clove::EditBox, [311](#)
- clove::EditComboBox, [334](#)
- clove::Expander, [361](#)
- clove::FlatView, [398](#)
- clove::Form, [420](#)
- clove::Grid, [441](#)
- clove::HorizontalStack, [468](#)
- clove::HtmlView, [490](#)
- clove::IconView, [513](#)
- clove::ImageView, [533](#)
- clove::Label, [554](#)
- clove::ListView, [582](#)
- clove::MainView, [611](#)
- clove::MediaPlayer, [638](#)
- clove::Menubar, [661](#)
- clove::ModalPanel, [682](#)
- clove::MultilineEditBox, [703](#)
- clove::NumericEditBox, [746](#)
- clove::PasswordEditBox, [771](#)
- clove::PopupMenu, [794](#)
- clove::PopupMenuButton, [816](#)
- clove::ProgressBar, [838](#)
- clove::RadioButton, [867](#)
- clove::RawResizeSplitter, [891](#)
- clove::ResizeSplitter, [912](#)
- clove::RichEdit, [935](#)
- clove::RichEditBox, [955](#)
- clove::ScrollView, [979](#)
- clove::Slider, [1001](#)
- clove::Spacer, [1032](#)
- clove::TabView, [1087](#)
- clove::TableView, [1060](#)
- clove::TimeBox, [1111](#)
- clove::ToolBar, [1135](#)
- clove::TreeView, [1161](#)
- clove::VerticalStack, [1200](#)
- clove::VideoPlayer, [1227](#)
- clove::Widget, [1250](#)
- clove::Wrap, [1270](#)
- OnLoaded
 - clove::ImageView, [533](#)
- OnLoadingAborted
 - clove::AudioPlayer, [78](#)
 - clove::MediaPlayer, [638](#)
 - clove::VideoPlayer, [1227](#)
- OnMainResized
 - clove::utils, [1184](#)
- OnMove
 - clove::RawResizeSplitter, [891](#)
- OnMoveBegin
 - clove::RawResizeSplitter, [891](#)
- OnMoveEnd
 - clove::RawResizeSplitter, [891](#)
- OnPopupTextSelected
 - clove::DateBox, [242](#)
 - clove::DropDownBox, [287](#)
 - clove::EditBox, [311](#)
 - clove::EditComboBox, [334](#)
 - clove::MultilineEditBox, [703](#)
 - clove::NumericEditBox, [747](#)
 - clove::PasswordEditBox, [771](#)
 - clove::TimeBox, [1111](#)
- OnPropertyChanged
 - clove::AbstractMenu, [36](#)
 - clove::AudioPlayer, [78](#)
 - clove::Border, [101](#)
 - clove::Button, [122](#)
 - clove::Carousel, [144](#)
 - clove::CheckButton, [169](#)
 - clove::DataView, [214](#)
 - clove::DateBox, [243](#)
 - clove::Dialog, [265](#)
 - clove::DropDownBox, [288](#)
 - clove::EditBox, [312](#)
 - clove::EditComboBox, [335](#)
 - clove::Expander, [361](#)
 - clove::FlatView, [398](#)
 - clove::Form, [420](#)
 - clove::Grid, [441](#)
 - clove::HorizontalStack, [468](#)
 - clove::HtmlView, [490](#)
 - clove::IconView, [513](#)
 - clove::ImageView, [534](#)
 - clove::Label, [554](#)
 - clove::ListView, [582](#)
 - clove::MainView, [611](#)
 - clove::MediaPlayer, [638](#)
 - clove::Menubar, [662](#)
 - clove::ModalPanel, [682](#)
 - clove::MultilineEditBox, [704](#)
 - clove::NumericEditBox, [747](#)
 - clove::PasswordEditBox, [772](#)
 - clove::PopupMenu, [795](#)
 - clove::PopupMenuButton, [816](#)
 - clove::ProgressBar, [838](#)
 - clove::RadioButton, [867](#)
 - clove::RawResizeSplitter, [891](#)
 - clove::ResizeSplitter, [912](#)
 - clove::RichEdit, [935](#)
 - clove::RichEditBox, [955](#)
 - clove::ScrollView, [979](#)
 - clove::Slider, [1001](#)
 - clove::Spacer, [1032](#)
 - clove::TabView, [1087](#)
 - clove::TableView, [1060](#)
 - clove::TimeBox, [1112](#)
 - clove::ToolBar, [1135](#)
 - clove::TreeView, [1161](#)
 - clove::VerticalStack, [1200](#)
 - clove::VideoPlayer, [1227](#)
 - clove::Widget, [1250](#)
 - clove::Wrap, [1270](#)
- OnReadyForPlayback
 - clove::AudioPlayer, [78](#)

- clove::MediaPlayer, 638
- clove::VideoPlayer, 1227
- OnResized
 - clove::AbstractMenu, 36
 - clove::AudioPlayer, 78
 - clove::Border, 101
 - clove::Button, 122
 - clove::Carousel, 144
 - clove::CheckBox, 169
 - clove::DataView, 214
 - clove::DateBox, 243
 - clove::Dialog, 265
 - clove::DropDownBox, 288
 - clove::EditBox, 312
 - clove::EditComboBox, 335
 - clove::Expander, 361
 - clove::FlatView, 398
 - clove::Form, 420
 - clove::Grid, 441
 - clove::HorizontalStack, 469
 - clove::HtmlView, 490
 - clove::IconView, 513
 - clove::ImageView, 534
 - clove::Label, 555
 - clove::ListView, 582
 - clove::MainView, 611
 - clove::MediaPlayer, 638
 - clove::Menubar, 662
 - clove::ModalPanel, 683
 - clove::MultilineEditBox, 704
 - clove::NumericEditBox, 747
 - clove::PasswordEditBox, 772
 - clove::PopupMenu, 795
 - clove::PopupMenuButton, 817
 - clove::ProgressBar, 838
 - clove::RadioButton, 867
 - clove::RawResizeSplitter, 892
 - clove::ResizeSplitter, 912
 - clove::RichEdit, 935
 - clove::RichEditBox, 955
 - clove::ScrollView, 979
 - clove::Slider, 1002
 - clove::Spacer, 1032
 - clove::TabView, 1087
 - clove::TableView, 1060
 - clove::TimeBox, 1112
 - clove::Toolbar, 1135
 - clove::TreeView, 1161
 - clove::VerticalStack, 1200
 - clove::VideoPlayer, 1227
 - clove::Widget, 1250
 - clove::Wrap, 1270
- OnSelectionChanged
 - clove::DataView, 215
 - clove::ListView, 583
 - clove::TableView, 1061
 - clove::TreeView, 1162
- OnTabCreationRequested
 - clove::Carousel, 145
 - clove::TabView, 1088
- OnVisibilityChanged
 - clove::AbstractMenu, 36
 - clove::AudioPlayer, 78
 - clove::Border, 101
 - clove::Button, 122
 - clove::Carousel, 145
 - clove::CheckBox, 169
 - clove::DataView, 215
 - clove::DateBox, 243
 - clove::Dialog, 265
 - clove::DropDownBox, 288
 - clove::EditBox, 312
 - clove::EditComboBox, 335
 - clove::Expander, 362
 - clove::FlatView, 398
 - clove::Form, 420
 - clove::Grid, 441
 - clove::HorizontalStack, 469
 - clove::HtmlView, 490
 - clove::IconView, 513
 - clove::ImageView, 534
 - clove::Label, 555
 - clove::ListView, 583
 - clove::MainView, 611
 - clove::MediaPlayer, 638
 - clove::Menubar, 662
 - clove::ModalPanel, 683
 - clove::MultilineEditBox, 704
 - clove::NumericEditBox, 747
 - clove::PasswordEditBox, 772
 - clove::PopupMenu, 795
 - clove::PopupMenuButton, 817
 - clove::ProgressBar, 838
 - clove::RadioButton, 867
 - clove::RawResizeSplitter, 892
 - clove::ResizeSplitter, 912
 - clove::RichEdit, 936
 - clove::RichEditBox, 956
 - clove::ScrollView, 979
 - clove::Slider, 1002
 - clove::Spacer, 1032
 - clove::TabView, 1088
 - clove::TableView, 1061
 - clove::TimeBox, 1112
 - clove::Toolbar, 1136
 - clove::TreeView, 1162
 - clove::VerticalStack, 1200
 - clove::VideoPlayer, 1227
 - clove::Widget, 1251
 - clove::Wrap, 1270
- orientation
 - clove::ProgressBar, 839
 - clove::ResizeSplitter, 913
 - clove::Slider, 1002
- outerSize
 - clove::AbstractMenu, 36

- clove::AudioPlayer, 78
- clove::Border, 101
- clove::Button, 122
- clove::Carousel, 145
- clove::CheckBox, 169
- clove::DataView, 215
- clove::DateBox, 243
- clove::Dialog, 266
- clove::DropDownBox, 288
- clove::EditBox, 312
- clove::EditComboBox, 335
- clove::Expander, 362
- clove::FlatView, 398
- clove::Form, 421
- clove::Grid, 441
- clove::HorizontalStack, 469
- clove::HtmlView, 491
- clove::IconView, 513
- clove::ImageView, 534
- clove::Label, 555
- clove::ListView, 583
- clove::MainView, 611
- clove::MediaPlayer, 639
- clove::Menubar, 662
- clove::ModalPanel, 683
- clove::MultilineEditBox, 704
- clove::NumericEditBox, 747
- clove::PasswordEditBox, 772
- clove::PopupMenu, 795
- clove::PopupMenuButton, 817
- clove::ProgressBar, 839
- clove::RadioButton, 867
- clove::RawResizeSplitter, 892
- clove::ResizeSplitter, 913
- clove::RichEdit, 936
- clove::RichEditBox, 956
- clove::ScrollView, 980
- clove::Slider, 1002
- clove::Spacer, 1033
- clove::TabView, 1088
- clove::TableView, 1061
- clove::TimeBox, 1112
- clove::Toolbar, 1136
- clove::TreeView, 1162
- clove::VerticalStack, 1200
- clove::VideoPlayer, 1228
- clove::Widget, 1251
- clove::Wrap, 1271
- parent
 - clove::AjaxAsyncDatasource, 53
 - clove::AsyncDatasource, 61
 - clove::Datasource, 192
 - clove::DatasourceValuePointer, 196
 - clove::FilterProxyDatasource, 377
 - clove::NativeDatasource, 722
 - clove::ProxyDatasource, 852
 - clove::SortProxyDatasource, 1018
- parentWidget
- clove::AbstractMenu, 36
- clove::AudioPlayer, 79
- clove::Border, 102
- clove::Button, 122
- clove::Carousel, 145
- clove::CheckBox, 169
- clove::DataView, 215
- clove::DateBox, 243
- clove::Dialog, 266
- clove::DropDownBox, 288
- clove::EditBox, 312
- clove::EditComboBox, 335
- clove::Expander, 362
- clove::FlatView, 399
- clove::Form, 421
- clove::Grid, 442
- clove::HorizontalStack, 469
- clove::HtmlView, 491
- clove::IconView, 514
- clove::ImageView, 534
- clove::Label, 555
- clove::ListView, 583
- clove::MainView, 612
- clove::MediaPlayer, 639
- clove::Menubar, 662
- clove::ModalPanel, 683
- clove::MultilineEditBox, 704
- clove::NumericEditBox, 747
- clove::PasswordEditBox, 772
- clove::PopupMenu, 795
- clove::PopupMenuButton, 817
- clove::ProgressBar, 839
- clove::RadioButton, 868
- clove::RawResizeSplitter, 892
- clove::ResizeSplitter, 913
- clove::RichEdit, 936
- clove::RichEditBox, 956
- clove::ScrollView, 980
- clove::Slider, 1002
- clove::Spacer, 1033
- clove::TabView, 1088
- clove::TableView, 1061
- clove::TimeBox, 1112
- clove::Toolbar, 1136
- clove::TreeView, 1162
- clove::VerticalStack, 1201
- clove::VideoPlayer, 1228
- clove::Widget, 1251
- clove::Wrap, 1271
- parseLangCode
 - clove::I18N, 499
- pause
 - clove::AudioPlayer, 79
 - clove::Carousel, 145
 - clove::MediaPlayer, 639
 - clove::VideoPlayer, 1228
- play
 - clove::AudioPlayer, 79

- clove::Carousel, [145](#)
- clove::MediaPlayer, [639](#)
- clove::VideoPlayer, [1228](#)
- populateUI
 - clove, [185](#)
- popup
 - clove::utils, [1184](#)
- popupItems
 - clove::DropDownBox, [288](#)
 - clove::EditComboBox, [335](#)
- preload
 - clove::AudioPlayer, [79](#)
 - clove::MediaPlayer, [639](#)
 - clove::VideoPlayer, [1228](#)
- proxyPointerToNativePointer
 - clove::FilterProxyDatasource, [378](#)
 - clove::ProxyDatasource, [853](#)
 - clove::SortProxyDatasource, [1018](#)
- RadioGroup
 - clove::RadioGroup, [877](#)
- readOnly
 - clove::DateBox, [243](#)
 - clove::DropDownBox, [289](#)
 - clove::EditBox, [312](#)
 - clove::EditComboBox, [336](#)
 - clove::MultilineEditBox, [704](#)
 - clove::NumericEditBox, [748](#)
 - clove::PasswordEditBox, [772](#)
 - clove::TimeBox, [1112](#)
- refresh
 - clove::FilterProxyDatasource, [378](#)
 - clove::ProxyDatasource, [853](#)
 - clove::SortProxyDatasource, [1019](#)
- registerBusy
 - clove::AbstractMenu, [36](#)
 - clove::AudioPlayer, [79](#)
 - clove::Border, [102](#)
 - clove::Button, [123](#)
 - clove::Carousel, [146](#)
 - clove::CheckButton, [169](#)
 - clove::DataView, [215](#)
 - clove::DateBox, [244](#)
 - clove::Dialog, [266](#)
 - clove::DropDownBox, [289](#)
 - clove::EditBox, [313](#)
 - clove::EditComboBox, [336](#)
 - clove::Expander, [362](#)
 - clove::FlatView, [399](#)
 - clove::Form, [421](#)
 - clove::Grid, [442](#)
 - clove::HorizontalStack, [469](#)
 - clove::HtmlView, [491](#)
 - clove::IconView, [514](#)
 - clove::ImageView, [534](#)
 - clove::Label, [555](#)
 - clove::ListView, [583](#)
 - clove::MainView, [612](#)
 - clove::MediaPlayer, [639](#)
 - clove::MenuBar, [662](#)
 - clove::ModalPanel, [683](#)
 - clove::MultilineEditBox, [705](#)
 - clove::NumericEditBox, [748](#)
 - clove::PasswordEditBox, [773](#)
 - clove::PopupMenu, [795](#)
 - clove::PopupMenuButton, [817](#)
 - clove::ProgressBar, [839](#)
 - clove::RawResizeSplitter, [892](#)
 - clove::ResizeSplitter, [913](#)
 - clove::RichEdit, [936](#)
 - clove::RichEditBox, [956](#)
 - clove::ScrollView, [980](#)
 - clove::Slider, [1002](#)
 - clove::Spacer, [1033](#)
 - clove::TabView, [1088](#)
 - clove::TableView, [1061](#)
 - clove::TimeBox, [1113](#)
 - clove::ToolBar, [1136](#)
 - clove::TreeView, [1162](#)
 - clove::VerticalStack, [1201](#)
 - clove::VideoPlayer, [1228](#)
 - clove::Widget, [1251](#)
 - clove::Wrap, [1271](#)
- relayout
 - clove::AbstractMenu, [37](#)
 - clove::AudioPlayer, [79](#)
 - clove::Border, [102](#)
 - clove::Button, [123](#)
 - clove::Carousel, [146](#)
 - clove::CheckButton, [170](#)
 - clove::DataView, [215](#)
 - clove::DateBox, [244](#)
 - clove::Dialog, [266](#)
 - clove::DropDownBox, [289](#)
 - clove::EditBox, [313](#)
 - clove::EditComboBox, [336](#)
 - clove::Expander, [362](#)
 - clove::FlatView, [399](#)
 - clove::Form, [421](#)
 - clove::Grid, [442](#)
 - clove::HorizontalStack, [469](#)
 - clove::HtmlView, [491](#)
 - clove::IconView, [514](#)
 - clove::ImageView, [535](#)
 - clove::Label, [555](#)
 - clove::ListView, [583](#)
 - clove::MainView, [612](#)
 - clove::MediaPlayer, [640](#)
 - clove::MenuBar, [663](#)
 - clove::ModalPanel, [683](#)
 - clove::MultilineEditBox, [705](#)
 - clove::NumericEditBox, [748](#)
 - clove::PasswordEditBox, [773](#)
 - clove::PopupMenu, [796](#)
 - clove::PopupMenuButton, [817](#)
 - clove::ProgressBar, [839](#)

- clove::RadioButton, [868](#)
- clove::RawResizeSplitter, [892](#)
- clove::ResizeSplitter, [913](#)
- clove::RichEdit, [936](#)
- clove::RichEditBox, [956](#)
- clove::ScrollView, [980](#)
- clove::Slider, [1003](#)
- clove::Spacer, [1033](#)
- clove::TabView, [1088](#)
- clove::TableView, [1061](#)
- clove::TimeBox, [1113](#)
- clove::ToolBar, [1136](#)
- clove::TreeView, [1162](#)
- clove::VerticalStack, [1201](#)
- clove::VideoPlayer, [1229](#)
- clove::Widget, [1251](#)
- clove::Wrap, [1271](#)
- remove
 - clove::AbstractMenu, [37](#)
 - clove::AudioPlayer, [80](#)
 - clove::Border, [102](#)
 - clove::Button, [123](#)
 - clove::Carousel, [146](#)
 - clove::CheckBox, [170](#)
 - clove::DataView, [216](#)
 - clove::DateBox, [244](#)
 - clove::Dialog, [266](#)
 - clove::DropDownBox, [289](#)
 - clove::EditBox, [313](#)
 - clove::EditComboBox, [336](#)
 - clove::Expander, [362](#)
 - clove::FlatView, [399](#)
 - clove::Form, [421](#)
 - clove::Grid, [442](#)
 - clove::HorizontalStack, [470](#)
 - clove::HtmlView, [491](#)
 - clove::IconView, [514](#)
 - clove::ImageView, [535](#)
 - clove::Label, [556](#)
 - clove::ListView, [584](#)
 - clove::MainView, [612](#)
 - clove::MediaPlayer, [640](#)
 - clove::Menubar, [663](#)
 - clove::ModalPanel, [684](#)
 - clove::MultilineEditBox, [705](#)
 - clove::NumericEditBox, [748](#)
 - clove::PasswordEditBox, [773](#)
 - clove::PopupMenu, [796](#)
 - clove::PopupMenuButton, [818](#)
 - clove::ProgressBar, [839](#)
 - clove::RadioButton, [868](#)
 - clove::RawResizeSplitter, [893](#)
 - clove::ResizeSplitter, [913](#)
 - clove::RichEdit, [936](#)
 - clove::RichEditBox, [956](#)
 - clove::ScrollView, [980](#)
 - clove::Slider, [1003](#)
 - clove::Spacer, [1033](#)
 - clove::TabView, [1089](#)
 - clove::TableView, [1062](#)
 - clove::TimeBox, [1113](#)
 - clove::ToolBar, [1136](#)
 - clove::TreeView, [1163](#)
 - clove::VerticalStack, [1201](#)
 - clove::VideoPlayer, [1229](#)
 - clove::Widget, [1251](#)
 - clove::Wrap, [1271](#)
- removeColumn
 - clove::Grid, [443](#)
 - clove::GridLayout, [452](#)
 - clove::HorizontalStack, [470](#)
 - clove::NativeDatasource, [722](#)
 - clove::NativeDatasourceNode, [730](#)
 - clove::StackLayout, [1042](#)
 - clove::VerticalStack, [1202](#)
- removeHandler
 - clove::Event, [347](#)
- removeOnDoubleClickTapHandler
 - clove::utils, [1185](#)
- removeRow
 - clove::Grid, [443](#)
 - clove::GridLayout, [452](#)
 - clove::HorizontalStack, [470](#)
 - clove::NativeDatasource, [723](#)
 - clove::NativeDatasourceNode, [731](#)
 - clove::StackLayout, [1042](#)
 - clove::VerticalStack, [1202](#)
- removeStyleClass
 - clove::AbstractMenu, [37](#)
 - clove::AudioPlayer, [80](#)
 - clove::Border, [103](#)
 - clove::Button, [123](#)
 - clove::Carousel, [146](#)
 - clove::CheckBox, [170](#)
 - clove::DataView, [216](#)
 - clove::DateBox, [244](#)
 - clove::Dialog, [267](#)
 - clove::DropDownBox, [290](#)
 - clove::EditBox, [313](#)
 - clove::EditComboBox, [337](#)
 - clove::Expander, [363](#)
 - clove::FlatView, [400](#)
 - clove::Form, [422](#)
 - clove::Grid, [443](#)
 - clove::HorizontalStack, [471](#)
 - clove::HtmlView, [492](#)
 - clove::IconView, [515](#)
 - clove::ImageView, [535](#)
 - clove::Label, [556](#)
 - clove::ListView, [584](#)
 - clove::MainView, [613](#)
 - clove::MediaPlayer, [640](#)
 - clove::Menubar, [663](#)
 - clove::ModalPanel, [684](#)
 - clove::MultilineEditBox, [705](#)
 - clove::NumericEditBox, [749](#)

- clove::PasswordEditBox, 773
- clove::PopupMenu, 796
- clove::PopupMenuButton, 818
- clove::ProgressBar, 840
- clove::RadioButton, 869
- clove::RawResizeSplitter, 893
- clove::ResizeSplitter, 914
- clove::RichEdit, 937
- clove::RichEditBox, 957
- clove::ScrollView, 981
- clove::Slider, 1003
- clove::Spacer, 1034
- clove::TabView, 1089
- clove::TableView, 1062
- clove::TimeBox, 1113
- clove::Toolbar, 1137
- clove::TreeView, 1163
- clove::VerticalStack, 1202
- clove::VideoPlayer, 1229
- clove::Widget, 1252
- clove::Wrap, 1272
- resize
 - clove::AbstractMenu, 38
 - clove::AudioPlayer, 80
 - clove::Border, 103
 - clove::Button, 124
 - clove::Carousel, 147
 - clove::CheckBox, 171
 - clove::DataView, 216
 - clove::DateBox, 245
 - clove::Dialog, 267
 - clove::DropDownBox, 290
 - clove::EditBox, 314
 - clove::EditComboBox, 337
 - clove::Expander, 363
 - clove::FlatView, 400
 - clove::Form, 422
 - clove::Grid, 443
 - clove::HorizontalStack, 471
 - clove::HtmlView, 492
 - clove::IconView, 515
 - clove::ImageView, 536
 - clove::Label, 556
 - clove::ListView, 584
 - clove::MainView, 613
 - clove::MediaPlayer, 641
 - clove::Menubar, 664
 - clove::ModalPanel, 684
 - clove::MultilineEditBox, 706
 - clove::NumericEditBox, 749
 - clove::PasswordEditBox, 774
 - clove::PopupMenu, 797
 - clove::PopupMenuButton, 818
 - clove::ProgressBar, 840
 - clove::RadioButton, 869
 - clove::RawResizeSplitter, 893
 - clove::ResizeSplitter, 914
 - clove::RichEdit, 937
 - clove::RichEditBox, 957
 - clove::ScrollView, 981
 - clove::Slider, 1004
 - clove::Spacer, 1034
 - clove::TabView, 1089
 - clove::TableView, 1062
 - clove::TimeBox, 1114
 - clove::Toolbar, 1137
 - clove::TreeView, 1163
 - clove::VerticalStack, 1202
 - clove::VideoPlayer, 1230
 - clove::Widget, 1252
 - clove::Wrap, 1272
- resumeResizing
 - clove::utils, 1185
- richeditbox
 - clove::RichEdit, 937
- root
 - clove::NativeDatasource, 723
- rootNameScope
 - clove, 186
- rowCount
 - clove::AjaxAsyncDatasource, 53
 - clove::AsyncDatasource, 62
 - clove::Datasource, 192
 - clove::DatasourceValuePointer, 196
 - clove::FilterProxyDatasource, 378
 - clove::NativeDatasource, 723
 - clove::NativeDatasourceNode, 731
 - clove::ProxyDatasource, 853
 - clove::SortProxyDatasource, 1019
- rowHeadersVisible
 - clove::AjaxAsyncDatasource, 53
 - clove::AsyncDatasource, 62
 - clove::FilterProxyDatasource, 378
 - clove::Headersource, 455
 - clove::NativeDatasource, 723
 - clove::ProxyDatasource, 853
 - clove::SortProxyDatasource, 1019
- rows
 - clove::VerticalStack, 1203
- rowsResizable
 - clove::DataView, 216
 - clove::ListView, 584
 - clove::TableView, 1062
 - clove::TreeView, 1163
- runOnLoaded
 - clove::utils, 1185
- sections
 - clove::Form, 422
- select
 - clove::RadioGroup, 878
- selectByIndex
 - clove::RadioGroup, 878
- selectByValue
 - clove::RadioGroup, 879
- selectCell
 - clove::DataView, 217

- clove::ListView, 585
 - clove::TableView, 1063
 - clove::TreeView, 1164
- selected
 - clove::RadioGroup, 879
- selectedIndex
 - clove::RadioGroup, 879
- selectedValue
 - clove::RadioGroup, 879
- selection
 - clove::DataView, 217
 - clove::ListView, 585
 - clove::RichEditBox, 957
 - clove::TableView, 1063
 - clove::TreeView, 1164
- selectionDecreaseFontSize
 - clove::RichEditBox, 957
- selectionDialog
 - clove, 186
- selectionIncreaseFontSize
 - clove::RichEditBox, 957
- selectionInsertList
 - clove::RichEditBox, 958
- selectionSetBackgroundColor
 - clove::RichEditBox, 958
- selectionSetForegroundColor
 - clove::RichEditBox, 958
- setActions
 - clove::AbstractMenu, 38
 - clove::MainView, 613
 - clove::Menubar, 664
 - clove::PopupMenu, 797
 - clove::PopupMenuButton, 818
 - clove::Toolbar, 1137
- setAllowChecking
 - clove::DataView, 217
 - clove::ListView, 585
 - clove::TableView, 1063
 - clove::TreeView, 1164
- setAllowSelection
 - clove::DataView, 217
 - clove::ListView, 585
 - clove::TableView, 1063
 - clove::TreeView, 1164
- setAlwaysAllocateExpanderSpace
 - clove::DataView, 218
 - clove::ListView, 586
 - clove::TableView, 1064
 - clove::TreeView, 1165
- setAutoPlay
 - clove::AudioPlayer, 80
 - clove::MediaPlayer, 641
 - clove::VideoPlayer, 1230
- setAutocompletionFilter
 - clove::DateBox, 245
 - clove::DropDownBox, 290
 - clove::EditBox, 314
 - clove::EditComboBox, 337
- clove::MultilineEditBox, 706
 - clove::NumericEditBox, 749
 - clove::PasswordEditBox, 774
 - clove::TimeBox, 1114
- setAutocompletionItems
 - clove::DateBox, 245
 - clove::DropDownBox, 290
 - clove::EditBox, 314
 - clove::EditComboBox, 337
 - clove::MultilineEditBox, 706
 - clove::NumericEditBox, 749
 - clove::PasswordEditBox, 774
 - clove::TimeBox, 1114
- setAutocompletionOpenForNoText
 - clove::DateBox, 245
 - clove::DropDownBox, 291
 - clove::EditBox, 314
 - clove::EditComboBox, 338
 - clove::MultilineEditBox, 706
 - clove::NumericEditBox, 750
 - clove::PasswordEditBox, 774
 - clove::TimeBox, 1114
- setBody
 - clove::Border, 103
 - clove::Dialog, 267
 - clove::Expander, 363
 - clove::ResizeSplitter, 914
 - clove::ScrollView, 981
- setBodyLeft
 - clove::MainView, 613
- setBodyLeftViewActionLabel
 - clove::MainView, 614
- setBodyRight
 - clove::MainView, 614
- setBodyRightViewActionLabel
 - clove::MainView, 614
- setBodyWidget
 - clove::ResizeSplitter, 914
- setBusy
 - clove::AbstractMenu, 38
 - clove::AudioPlayer, 81
 - clove::Border, 103
 - clove::Button, 124
 - clove::Carousel, 147
 - clove::CheckButton, 171
 - clove::DataView, 218
 - clove::DateBox, 246
 - clove::Dialog, 267
 - clove::DropDownBox, 291
 - clove::EditBox, 315
 - clove::EditComboBox, 338
 - clove::Expander, 363
 - clove::FlatView, 400
 - clove::Form, 422
 - clove::Grid, 444
 - clove::HorizontalStack, 471
 - clove::HtmlView, 492
 - clove::IconView, 515

- clove::ImageView, 536
- clove::Label, 556
- clove::ListView, 586
- clove::MainView, 614
- clove::MediaPlayer, 641
- clove::Menubar, 664
- clove::ModalPanel, 684
- clove::MultilineEditBox, 707
- clove::NumericEditBox, 750
- clove::PasswordEditBox, 775
- clove::PopupMenu, 797
- clove::PopupMenuButton, 819
- clove::ProgressBar, 840
- clove::RadioButton, 869
- clove::RawResizeSplitter, 893
- clove::ResizeSplitter, 916
- clove::RichEdit, 937
- clove::RichEditBox, 959
- clove::ScrollView, 981
- clove::Slider, 1004
- clove::Spacer, 1034
- clove::TabView, 1089
- clove::TableView, 1064
- clove::TimeBox, 1115
- clove::ToolBar, 1137
- clove::TreeView, 1165
- clove::VerticalStack, 1203
- clove::VideoPlayer, 1230
- clove::Widget, 1252
- clove::Wrap, 1272
- setCellChecked
 - clove::DataView, 218
 - clove::ListView, 586
 - clove::TableView, 1064
 - clove::TreeView, 1165
- setCellGenerator
 - clove::DataView, 218
 - clove::ListView, 586
 - clove::TableView, 1064
 - clove::TreeView, 1165
- setCheckable
 - clove::Button, 124
 - clove::PopupMenuButton, 819
- setChecked
 - clove::Button, 124
 - clove::CheckButton, 171
 - clove::PopupMenuButton, 819
 - clove::RadioButton, 869
- setCheckedCells
 - clove::DataView, 219
 - clove::ListView, 587
 - clove::TableView, 1065
 - clove::TreeView, 1166
- setChildren
 - clove::FlatLayout, 382
 - clove::FlatView, 400
 - clove::Grid, 444
 - clove::GridLayout, 453
 - clove::HorizontalStack, 471
 - clove::Layout, 565
 - clove::StackLayout, 1043
 - clove::VerticalStack, 1203
 - clove::Wrap, 1272
 - clove::WrapLayout, 1280
- setCollapseHidden
 - clove::FlatLayout, 383
 - clove::FlatView, 401
- setCollapsedIcon
 - clove::Expander, 364
- setCollapsedLabel
 - clove::Expander, 364
- setCols
 - clove::HorizontalStack, 472
- setColumnComparator
 - clove::SortProxyDatasource, 1019
- setColumnFilter
 - clove::FilterProxyDatasource, 379
- setColumnHeader
 - clove::NativeDatasource, 724
- setColumnsResizable
 - clove::DataView, 219
 - clove::ListView, 587
 - clove::TableView, 1065
 - clove::TreeView, 1166
- setContentRoot
 - clove::HtmlView, 492
- setCurrentMediaPosition
 - clove::AudioPlayer, 81
 - clove::MediaPlayer, 641
 - clove::VideoPlayer, 1230
- setCurrentTab
 - clove::Carousel, 147
 - clove::TabView, 1090
- setCurrentView
 - clove::FlatLayout, 383
 - clove::FlatView, 401
- setDataViewProcessor
 - clove::DataView, 220
 - clove::ListView, 588
 - clove::TableView, 1066
 - clove::TreeView, 1167
- setDatasource
 - clove::DataView, 219
 - clove::FilterProxyDatasource, 379
 - clove::ListView, 587
 - clove::ProxyDatasource, 854
 - clove::SortProxyDatasource, 1020
 - clove::TableView, 1065
 - clove::TreeView, 1166
- setDate
 - clove::DateBox, 246
- setDoStandaloneResizing
 - clove::AbstractMenu, 38
 - clove::AudioPlayer, 81
 - clove::Border, 104
 - clove::Button, 125

- clove::Carousel, [147](#)
- clove::CheckBox, [171](#)
- clove::DataView, [220](#)
- clove::DateBox, [246](#)
- clove::Dialog, [268](#)
- clove::DropDownBox, [291](#)
- clove::EditBox, [315](#)
- clove::EditComboBox, [338](#)
- clove::Expander, [364](#)
- clove::FlatView, [401](#)
- clove::Form, [423](#)
- clove::Grid, [444](#)
- clove::HorizontalStack, [472](#)
- clove::HtmlView, [493](#)
- clove::IconView, [515](#)
- clove::ImageView, [536](#)
- clove::Label, [557](#)
- clove::ListView, [588](#)
- clove::MainView, [615](#)
- clove::MediaPlayer, [642](#)
- clove::Menubar, [664](#)
- clove::ModalPanel, [685](#)
- clove::MultilineEditBox, [707](#)
- clove::NumericEditBox, [750](#)
- clove::PasswordEditBox, [775](#)
- clove::PopupMenu, [797](#)
- clove::PopupMenuButton, [820](#)
- clove::ProgressBar, [840](#)
- clove::RadioButton, [870](#)
- clove::RawResizeSplitter, [894](#)
- clove::ResizeSplitter, [916](#)
- clove::RichEdit, [938](#)
- clove::RichEditBox, [959](#)
- clove::ScrollView, [982](#)
- clove::Slider, [1004](#)
- clove::Spacer, [1034](#)
- clove::TabView, [1090](#)
- clove::TableView, [1066](#)
- clove::TimeBox, [1115](#)
- clove::ToolBar, [1138](#)
- clove::TreeView, [1167](#)
- clove::VerticalStack, [1203](#)
- clove::VideoPlayer, [1231](#)
- clove::Widget, [1252](#)
- clove::Wrap, [1273](#)
- setEditOnGesture
 - clove::DataView, [220](#)
 - clove::ListView, [588](#)
 - clove::TableView, [1066](#)
 - clove::TreeView, [1167](#)
- setEnabled
 - clove::AbstractMenu, [39](#)
 - clove::AudioPlayer, [82](#)
 - clove::Border, [104](#)
 - clove::Button, [125](#)
 - clove::Carousel, [148](#)
 - clove::CheckBox, [172](#)
 - clove::DataView, [220](#)
 - clove::DateBox, [246](#)
 - clove::Dialog, [268](#)
 - clove::DropDownBox, [291](#)
 - clove::EditBox, [315](#)
 - clove::EditComboBox, [338](#)
 - clove::Expander, [364](#)
 - clove::FlatView, [401](#)
 - clove::Form, [423](#)
 - clove::Grid, [444](#)
 - clove::HorizontalStack, [472](#)
 - clove::HtmlView, [493](#)
 - clove::IconView, [515](#)
 - clove::ImageView, [536](#)
 - clove::Label, [557](#)
 - clove::ListView, [588](#)
 - clove::MainView, [615](#)
 - clove::MediaPlayer, [642](#)
 - clove::Menubar, [665](#)
 - clove::ModalPanel, [685](#)
 - clove::MultilineEditBox, [707](#)
 - clove::NumericEditBox, [750](#)
 - clove::PasswordEditBox, [775](#)
 - clove::PopupMenu, [798](#)
 - clove::PopupMenuButton, [820](#)
 - clove::ProgressBar, [841](#)
 - clove::RadioButton, [870](#)
 - clove::RawResizeSplitter, [894](#)
 - clove::ResizeSplitter, [916](#)
 - clove::RichEdit, [938](#)
 - clove::RichEditBox, [959](#)
 - clove::ScrollView, [982](#)
 - clove::Slider, [1004](#)
 - clove::Spacer, [1035](#)
 - clove::TabView, [1090](#)
 - clove::TableView, [1066](#)
 - clove::TimeBox, [1115](#)
 - clove::ToolBar, [1138](#)
 - clove::TreeView, [1167](#)
 - clove::VerticalStack, [1205](#)
 - clove::VideoPlayer, [1231](#)
 - clove::Widget, [1253](#)
 - clove::Wrap, [1273](#)
- setExpandedIcon
 - clove::Expander, [365](#)
- setExpandedLabel
 - clove::Expander, [365](#)
- setExpanderIndicatorDirection
 - clove::PopupMenu, [798](#)
- setFocusBuddy
 - clove::Label, [557](#)
- setFullyTransparent
 - clove::ModalPanel, [685](#)
- setGranularity
 - clove::DataView, [221](#)
 - clove::ListView, [589](#)
 - clove::TableView, [1067](#)
 - clove::TreeView, [1168](#)
- setGridVisible

- clove::DataView, 221
- clove::ListView, 589
- clove::TableView, 1067
- clove::TreeView, 1168
- setGroup
 - clove::RadioButton, 870
- setGroupValue
 - clove::RadioButton, 870
- setHead1
 - clove::MainView, 615
 - clove::Toolbar, 1138
- setHead2
 - clove::MainView, 616
 - clove::Toolbar, 1138
- setHeadControl
 - clove::MainView, 616
 - clove::Toolbar, 1139
- setHeadersource
 - clove::DataView, 221
 - clove::ListView, 589
 - clove::TableView, 1067
 - clove::TreeView, 1168
- setHideExpanders
 - clove::DataView, 222
 - clove::ListView, 590
 - clove::TableView, 1068
 - clove::TreeView, 1169
- setHintText
 - clove::DateBox, 247
 - clove::DropDownBox, 292
 - clove::EditBox, 315
 - clove::EditComboBox, 339
 - clove::MultilineEditBox, 707
 - clove::NumericEditBox, 751
 - clove::PasswordEditBox, 775
 - clove::TimeBox, 1115
- setHorizontalScrollPosition
 - clove::ScrollView, 982
- setHorizontalStretchAffinity
 - clove::AbstractMenu, 39
 - clove::AudioPlayer, 82
 - clove::Border, 104
 - clove::Button, 125
 - clove::Carousel, 148
 - clove::CheckButton, 172
 - clove::DataView, 222
 - clove::DateBox, 247
 - clove::Dialog, 268
 - clove::DropDownBox, 292
 - clove::EditBox, 316
 - clove::EditComboBox, 339
 - clove::Expander, 365
 - clove::FlatView, 402
 - clove::Form, 423
 - clove::Grid, 445
 - clove::HorizontalStack, 472
 - clove::HtmlView, 493
 - clove::IconView, 516
- clove::ImageView, 537
- clove::Label, 558
- clove::ListView, 590
- clove::MainView, 616
- clove::MediaPlayer, 642
- clove::Menubar, 665
- clove::ModalPanel, 686
- clove::MultilineEditBox, 708
- clove::NumericEditBox, 751
- clove::PasswordEditBox, 776
- clove::PopupMenu, 798
- clove::PopupMenuButton, 820
- clove::ProgressBar, 841
- clove::RadioButton, 871
- clove::RawResizeSplitter, 894
- clove::ResizeSplitter, 916
- clove::RichEdit, 938
- clove::RichEditBox, 959
- clove::ScrollView, 982
- clove::Slider, 1005
- clove::Spacer, 1035
- clove::TabView, 1091
- clove::TableView, 1068
- clove::TimeBox, 1116
- clove::Toolbar, 1139
- clove::TreeView, 1169
- clove::VerticalStack, 1205
- clove::VideoPlayer, 1231
- clove::Widget, 1253
- clove::Wrap, 1273
- setHstretch
 - clove::AbstractMenu, 39
 - clove::AudioPlayer, 82
 - clove::Border, 104
 - clove::Button, 125
 - clove::Carousel, 148
 - clove::CheckButton, 172
 - clove::DataView, 222
 - clove::DateBox, 247
 - clove::Dialog, 268
 - clove::DropDownBox, 292
 - clove::EditBox, 316
 - clove::EditComboBox, 339
 - clove::Expander, 366
 - clove::FlatView, 402
 - clove::Form, 423
 - clove::Grid, 445
 - clove::HorizontalStack, 473
 - clove::HtmlView, 493
 - clove::IconView, 516
 - clove::ImageView, 537
 - clove::Label, 558
 - clove::ListView, 590
 - clove::MainView, 616
 - clove::MediaPlayer, 642
 - clove::Menubar, 665
 - clove::ModalPanel, 686
 - clove::MultilineEditBox, 708

- clove::NumericEditBox, 751
- clove::PasswordEditBox, 776
- clove::PopupMenu, 798
- clove::PopupMenuButton, 820
- clove::ProgressBar, 841
- clove::RadioButton, 871
- clove::RawResizeSplitter, 895
- clove::ResizeSplitter, 917
- clove::RichEdit, 938
- clove::RichEditBox, 960
- clove::ScrollView, 983
- clove::Slider, 1005
- clove::Spacer, 1035
- clove::TabView, 1091
- clove::TableView, 1068
- clove::TimeBox, 1116
- clove::ToolBar, 1139
- clove::TreeView, 1169
- clove::VerticalStack, 1205
- clove::VideoPlayer, 1231
- clove::Widget, 1253
- clove::Wrap, 1273
- setHtmlContent
 - clove::Label, 558
 - clove::RichEdit, 939
 - clove::RichEditBox, 960
- setIcon
 - clove::Button, 126
 - clove::IconView, 516
 - clove::MainView, 617
 - clove::PopupMenuButton, 821
 - clove::ToolBar, 1140
- setInterval
 - clove::Carousel, 148
- setIsExpanded
 - clove::Expander, 366
- setKeepAspectRatio
 - clove::ImageView, 537
- setLabel
 - clove::Button, 126
 - clove::CheckButton, 172
 - clove::Label, 558
 - clove::PopupMenuButton, 821
 - clove::ProgressBar, 841
 - clove::RadioButton, 871
- setLabelFunction
 - clove::ProgressBar, 842
- setLanguage
 - clove::l18N, 500
- setLoop
 - clove::AudioPlayer, 82
 - clove::MediaPlayer, 643
 - clove::VideoPlayer, 1232
- setMax
 - clove::NumericEditBox, 752
 - clove::Slider, 1005
- setMayFocus
 - clove::AbstractMenu, 39
- clove::AudioPlayer, 83
- clove::Border, 105
- clove::Button, 126
- clove::Carousel, 149
- clove::CheckButton, 173
- clove::DataView, 222
- clove::DateBox, 248
- clove::Dialog, 269
- clove::DropDownBox, 293
- clove::EditBox, 316
- clove::EditComboBox, 340
- clove::Expander, 366
- clove::FlatView, 402
- clove::Form, 424
- clove::Grid, 445
- clove::HorizontalStack, 473
- clove::HtmlView, 494
- clove::IconView, 517
- clove::ImageView, 537
- clove::Label, 559
- clove::ListView, 590
- clove::MainView, 617
- clove::MediaPlayer, 643
- clove::Menubar, 665
- clove::ModalPanel, 686
- clove::MultilineEditBox, 708
- clove::NumericEditBox, 752
- clove::PasswordEditBox, 776
- clove::PopupMenu, 799
- clove::PopupMenuButton, 821
- clove::ProgressBar, 842
- clove::RadioButton, 872
- clove::RawResizeSplitter, 895
- clove::ResizeSplitter, 917
- clove::RichEdit, 939
- clove::RichEditBox, 960
- clove::ScrollView, 983
- clove::Slider, 1005
- clove::Spacer, 1035
- clove::TabView, 1091
- clove::TableView, 1068
- clove::TimeBox, 1116
- clove::ToolBar, 1140
- clove::TreeView, 1169
- clove::VerticalStack, 1205
- clove::VideoPlayer, 1232
- clove::Widget, 1253
- clove::Wrap, 1274
- setMetadata
 - clove::NativeDatasource, 724
 - clove::NativeDatasourceNode, 731
- setMin
 - clove::NumericEditBox, 752
 - clove::Slider, 1006
- setMuted
 - clove::AudioPlayer, 83
 - clove::MediaPlayer, 643
 - clove::VideoPlayer, 1232

setName

clove::AbstractMenu, 40
clove::AudioPlayer, 83
clove::Border, 105
clove::Button, 126
clove::Carousel, 149
clove::CheckBox, 173
clove::DataView, 223
clove::DateBox, 248
clove::Dialog, 269
clove::DropDownBox, 293
clove::EditBox, 317
clove::EditComboBox, 340
clove::Expander, 366
clove::FlatView, 403
clove::Form, 424
clove::Grid, 446
clove::HorizontalStack, 473
clove::HtmlView, 494
clove::IconView, 517
clove::ImageView, 538
clove::Label, 559
clove::ListView, 591
clove::MainView, 617
clove::MediaPlayer, 643
clove::Menubar, 666
clove::ModalPanel, 686
clove::MultilineEditBox, 709
clove::NumericEditBox, 752
clove::PasswordEditBox, 777
clove::PopupMenu, 799
clove::PopupMenuButton, 822
clove::ProgressBar, 842
clove::RadioButton, 872
clove::RawResizeSplitter, 895
clove::ResizeSplitter, 917
clove::RichEdit, 939
clove::RichEditBox, 960
clove::ScrollView, 983
clove::Slider, 1006
clove::Spacer, 1036
clove::TabView, 1091
clove::TableView, 1069
clove::TimeBox, 1117
clove::Toolbar, 1140
clove::TreeView, 1170
clove::VerticalStack, 1206
clove::VideoPlayer, 1232
clove::Widget, 1254
clove::Wrap, 1274

setNodeActivationNeedsDoubleClick

clove::DataView, 223
clove::ListView, 591
clove::TableView, 1069
clove::TreeView, 1170

setOrientation

clove::ProgressBar, 843
clove::ResizeSplitter, 918

clove::Slider, 1006

setPopupsItems

clove::DropDownBox, 293
clove::EditComboBox, 340

setPreload

clove::AudioPlayer, 84
clove::MediaPlayer, 644
clove::VideoPlayer, 1233

setProperty

clove::AbstractMenu, 40
clove::AudioPlayer, 84
clove::Border, 105
clove::Button, 127
clove::Carousel, 149
clove::CheckBox, 173
clove::DataView, 223
clove::DateBox, 248
clove::Dialog, 269
clove::DropDownBox, 293
clove::EditBox, 317
clove::EditComboBox, 340
clove::Expander, 367
clove::FlatView, 403
clove::Form, 424
clove::Grid, 446
clove::HorizontalStack, 474
clove::HtmlView, 494
clove::IconView, 517
clove::ImageView, 538
clove::Label, 559
clove::ListView, 591
clove::MainView, 618
clove::MediaPlayer, 644
clove::Menubar, 666
clove::ModalPanel, 687
clove::MultilineEditBox, 709
clove::NumericEditBox, 753
clove::PasswordEditBox, 777
clove::PopupMenu, 799
clove::PopupMenuButton, 822
clove::ProgressBar, 843
clove::RadioButton, 872
clove::RawResizeSplitter, 895
clove::ResizeSplitter, 918
clove::RichEdit, 939
clove::RichEditBox, 961
clove::ScrollView, 984
clove::Slider, 1006
clove::Spacer, 1036
clove::TabView, 1092
clove::TableView, 1069
clove::TimeBox, 1117
clove::Toolbar, 1140
clove::TreeView, 1170
clove::VerticalStack, 1206
clove::VideoPlayer, 1233
clove::Widget, 1254
clove::Wrap, 1274

- setReadOnly
 - clove::DateBox, 249
 - clove::DropDownBox, 294
 - clove::EditBox, 317
 - clove::EditComboBox, 341
 - clove::MultilineEditBox, 709
 - clove::NumericEditBox, 753
 - clove::PasswordEditBox, 777
 - clove::TimeBox, 1117
- setRootValue
 - clove::NativeDatasource, 724
- setRowComparator
 - clove::SortProxyDatasource, 1020
- setRowFilter
 - clove::FilterProxyDatasource, 379
- setRowHeader
 - clove::NativeDatasource, 725
- setRows
 - clove::VerticalStack, 1206
- setRowsResizable
 - clove::DataView, 224
 - clove::ListView, 592
 - clove::TableView, 1070
 - clove::TreeView, 1171
- setSections
 - clove::Form, 425
- setSelection
 - clove::RichEditBox, 961
- setShowChangeMenu
 - clove::DataView, 224
 - clove::ListView, 592
 - clove::TableView, 1070
 - clove::TreeView, 1171
- setShowControls
 - clove::AudioPlayer, 84
 - clove::MediaPlayer, 644
 - clove::VideoPlayer, 1233
- setShowOnly
 - clove::MainView, 618
 - clove::ResizeSplitter, 918
- setShowOnlyFirstColumn
 - clove::DataView, 224
 - clove::ListView, 592
 - clove::TableView, 1070
 - clove::TreeView, 1171
- setSidebar
 - clove::MainView, 618
- setSize
 - clove::IconView, 518
- setSizeFractions
 - clove::ResizeSplitter, 919
- setSource
 - clove::AudioPlayer, 85
 - clove::ImageView, 538
 - clove::MediaPlayer, 645
 - clove::VideoPlayer, 1234
- setSplitterPosition
 - clove::MainView, 619
- setSplitterVisibility
 - clove::ResizeSplitter, 919
- setSplitterWidth
 - clove::ResizeSplitter, 919
- setStepsize
 - clove::NumericEditBox, 753
 - clove::Slider, 1007
- setStrictHorizontalSizing
 - clove::AbstractMenu, 40
 - clove::AudioPlayer, 85
 - clove::Border, 106
 - clove::Button, 127
 - clove::Carousel, 150
 - clove::CheckButton, 174
 - clove::DataView, 224
 - clove::DateBox, 249
 - clove::Dialog, 270
 - clove::DropDownBox, 294
 - clove::EditBox, 318
 - clove::EditComboBox, 341
 - clove::Expander, 367
 - clove::FlatView, 403
 - clove::Form, 425
 - clove::Grid, 446
 - clove::HorizontalStack, 474
 - clove::HtmlView, 495
 - clove::IconView, 518
 - clove::ImageView, 539
 - clove::Label, 560
 - clove::ListView, 592
 - clove::MainView, 619
 - clove::MediaPlayer, 645
 - clove::Menubar, 666
 - clove::ModalPanel, 687
 - clove::MultilineEditBox, 710
 - clove::NumericEditBox, 754
 - clove::PasswordEditBox, 778
 - clove::PopupMenu, 800
 - clove::PopupMenuButton, 822
 - clove::ProgressBar, 843
 - clove::RadioButton, 873
 - clove::RawResizeSplitter, 896
 - clove::ResizeSplitter, 919
 - clove::RichEdit, 940
 - clove::RichEditBox, 961
 - clove::ScrollView, 984
 - clove::Slider, 1007
 - clove::Spacer, 1036
 - clove::TabView, 1092
 - clove::TableView, 1070
 - clove::TimeBox, 1118
 - clove::Toolbar, 1141
 - clove::TreeView, 1171
 - clove::VerticalStack, 1207
 - clove::VideoPlayer, 1234
 - clove::Widget, 1254
 - clove::Wrap, 1275
- setStrictVerticalSizing

- clove::AbstractMenu, [41](#)
- clove::AudioPlayer, [85](#)
- clove::Border, [106](#)
- clove::Button, [127](#)
- clove::Carousel, [150](#)
- clove::CheckBox, [174](#)
- clove::DataView, [225](#)
- clove::DateBox, [249](#)
- clove::Dialog, [270](#)
- clove::DropDownBox, [294](#)
- clove::EditBox, [318](#)
- clove::EditComboBox, [341](#)
- clove::Expander, [367](#)
- clove::FlatView, [404](#)
- clove::Form, [425](#)
- clove::Grid, [447](#)
- clove::HorizontalStack, [474](#)
- clove::HtmlView, [495](#)
- clove::IconView, [518](#)
- clove::ImageView, [539](#)
- clove::Label, [560](#)
- clove::ListView, [593](#)
- clove::MainView, [619](#)
- clove::MediaPlayer, [645](#)
- clove::Menubar, [667](#)
- clove::ModalPanel, [687](#)
- clove::MultilineEditBox, [710](#)
- clove::NumericEditBox, [754](#)
- clove::PasswordEditBox, [778](#)
- clove::PopupMenu, [800](#)
- clove::PopupMenuButton, [823](#)
- clove::ProgressBar, [844](#)
- clove::RadioButton, [873](#)
- clove::RawResizeSplitter, [896](#)
- clove::ResizeSplitter, [920](#)
- clove::RichEdit, [940](#)
- clove::RichEditBox, [962](#)
- clove::ScrollView, [984](#)
- clove::Slider, [1007](#)
- clove::Spacer, [1037](#)
- clove::TabView, [1092](#)
- clove::TableView, [1071](#)
- clove::TimeBox, [1118](#)
- clove::Toolbar, [1141](#)
- clove::TreeView, [1172](#)
- clove::VerticalStack, [1207](#)
- clove::VideoPlayer, [1234](#)
- clove::Widget, [1255](#)
- clove::Wrap, [1275](#)
- clove::Dialog, [270](#)
- clove::DropDownBox, [295](#)
- clove::EditBox, [318](#)
- clove::EditComboBox, [342](#)
- clove::Expander, [368](#)
- clove::FlatView, [404](#)
- clove::Form, [425](#)
- clove::Grid, [447](#)
- clove::HorizontalStack, [475](#)
- clove::HtmlView, [495](#)
- clove::IconView, [518](#)
- clove::ImageView, [539](#)
- clove::Label, [560](#)
- clove::ListView, [593](#)
- clove::MainView, [619](#)
- clove::MediaPlayer, [645](#)
- clove::Menubar, [667](#)
- clove::ModalPanel, [688](#)
- clove::MultilineEditBox, [710](#)
- clove::NumericEditBox, [754](#)
- clove::PasswordEditBox, [778](#)
- clove::PopupMenu, [800](#)
- clove::PopupMenuButton, [823](#)
- clove::ProgressBar, [844](#)
- clove::RadioButton, [873](#)
- clove::RawResizeSplitter, [896](#)
- clove::ResizeSplitter, [920](#)
- clove::RichEdit, [940](#)
- clove::RichEditBox, [962](#)
- clove::ScrollView, [985](#)
- clove::Slider, [1008](#)
- clove::Spacer, [1037](#)
- clove::TabView, [1093](#)
- clove::TableView, [1071](#)
- clove::TimeBox, [1118](#)
- clove::Toolbar, [1141](#)
- clove::TreeView, [1172](#)
- clove::VerticalStack, [1207](#)
- clove::VideoPlayer, [1234](#)
- clove::Widget, [1255](#)
- clove::Wrap, [1275](#)
- setStyleClass
 - clove::AbstractMenu, [41](#)
 - clove::AudioPlayer, [86](#)
 - clove::Border, [106](#)
 - clove::Button, [128](#)
 - clove::Carousel, [150](#)
 - clove::CheckBox, [174](#)
 - clove::DataView, [225](#)
 - clove::DateBox, [250](#)
 - clove::Dialog, [270](#)
 - clove::DropDownBox, [295](#)
 - clove::EditBox, [318](#)
 - clove::EditComboBox, [342](#)
 - clove::Expander, [368](#)
 - clove::FlatView, [404](#)
 - clove::Form, [426](#)
 - clove::Grid, [447](#)

- clove::HorizontalStack, 475
- clove::HtmlView, 495
- clove::IconView, 519
- clove::ImageView, 539
- clove::Label, 560
- clove::ListView, 593
- clove::MainView, 620
- clove::MediaPlayer, 646
- clove::Menubar, 667
- clove::ModalPanel, 688
- clove::MultilineEditBox, 710
- clove::NumericEditBox, 754
- clove::PasswordEditBox, 778
- clove::PopupMenu, 800
- clove::PopupMenuButton, 823
- clove::ProgressBar, 844
- clove::RadioButton, 873
- clove::RawResizeSplitter, 897
- clove::ResizeSplitter, 920
- clove::RichEdit, 941
- clove::RichEditBox, 962
- clove::ScrollView, 985
- clove::Slider, 1008
- clove::Spacer, 1037
- clove::TabView, 1093
- clove::TableView, 1071
- clove::TimeBox, 1118
- clove::Toolbar, 1142
- clove::TreeView, 1172
- clove::VerticalStack, 1207
- clove::VideoPlayer, 1235
- clove::Widget, 1255
- clove::Wrap, 1275
- setStyleClassAssigned
 - clove::AbstractMenu, 41
 - clove::AudioPlayer, 86
 - clove::Border, 107
 - clove::Button, 128
 - clove::Carousel, 151
 - clove::CheckBox, 175
 - clove::DataView, 226
 - clove::DateBox, 250
 - clove::Dialog, 271
 - clove::DropDownBox, 295
 - clove::EditBox, 319
 - clove::EditComboBox, 342
 - clove::Expander, 368
 - clove::FlatView, 404
 - clove::Form, 426
 - clove::Grid, 447
 - clove::HorizontalStack, 475
 - clove::HtmlView, 496
 - clove::IconView, 519
 - clove::ImageView, 540
 - clove::Label, 561
 - clove::ListView, 594
 - clove::MainView, 620
 - clove::MediaPlayer, 646
 - clove::Menubar, 667
 - clove::ModalPanel, 688
 - clove::MultilineEditBox, 711
 - clove::NumericEditBox, 755
 - clove::PasswordEditBox, 779
 - clove::PopupMenu, 801
 - clove::PopupMenuButton, 823
 - clove::ProgressBar, 844
 - clove::RadioButton, 874
 - clove::RawResizeSplitter, 897
 - clove::ResizeSplitter, 920
 - clove::RichEdit, 941
 - clove::RichEditBox, 962
 - clove::ScrollView, 985
 - clove::Slider, 1008
 - clove::Spacer, 1037
 - clove::TabView, 1093
 - clove::TableView, 1072
 - clove::TimeBox, 1119
 - clove::Toolbar, 1142
 - clove::TreeView, 1173
 - clove::VerticalStack, 1208
 - clove::VideoPlayer, 1235
 - clove::Widget, 1255
 - clove::Wrap, 1276
 - clove::utils, 1186
- setSwitchInvisibleAnimationDuration
 - clove::Carousel, 151
 - clove::FlatLayout, 383
 - clove::FlatView, 405
 - clove::TabView, 1093
- setSwitchInvisibleAnimationName
 - clove::Carousel, 151
 - clove::FlatLayout, 383
 - clove::FlatView, 405
 - clove::TabView, 1094
- setSwitchVisibleAnimationDuration
 - clove::Carousel, 152
 - clove::FlatLayout, 384
 - clove::FlatView, 405
 - clove::TabView, 1094
- setSwitchVisibleAnimationName
 - clove::Carousel, 152
 - clove::FlatLayout, 384
 - clove::FlatView, 405
 - clove::TabView, 1094
- setTabBarLocation
 - clove::Carousel, 152
 - clove::TabView, 1095
- setTabs
 - clove::Carousel, 152
 - clove::TabView, 1095
- setText
 - clove::DateBox, 250
 - clove::DropDownBox, 295
 - clove::EditBox, 319
 - clove::EditComboBox, 342
 - clove::MultilineEditBox, 711

- clove::NumericEditBox, 755
- clove::PasswordEditBox, 779
- clove::TimeBox, 1119
- setTextSelection
 - clove::DateBox, 250
 - clove::DropDownBox, 296
 - clove::EditBox, 319
 - clove::EditComboBox, 343
 - clove::MultilineEditBox, 711
 - clove::NumericEditBox, 755
 - clove::PasswordEditBox, 779
 - clove::TimeBox, 1119
- setTime
 - clove::TimeBox, 1120
- setTitle
 - clove::Dialog, 271
- setTitleIcon
 - clove::Dialog, 271
- setUserMayAddTabs
 - clove::Carousel, 153
 - clove::TabView, 1095
- setValue
 - clove::NativeDatasource, 725
 - clove::NativeDatasourceNode, 731
 - clove::NumericEditBox, 756
 - clove::ProgressBar, 845
 - clove::Slider, 1008
- setValueDef
 - clove::RadioButton, 874
- setVerticalScrollPosition
 - clove::ScrollView, 985
- setVerticalStretchAffinity
 - clove::AbstractMenu, 42
 - clove::AudioPlayer, 86
 - clove::Border, 107
 - clove::Button, 128
 - clove::Carousel, 153
 - clove::CheckButton, 175
 - clove::DataView, 226
 - clove::DateBox, 251
 - clove::Dialog, 272
 - clove::DropDownBox, 296
 - clove::EditBox, 320
 - clove::EditComboBox, 343
 - clove::Expander, 368
 - clove::FlatView, 406
 - clove::Form, 426
 - clove::Grid, 448
 - clove::HorizontalStack, 475
 - clove::HtmlView, 496
 - clove::IconView, 519
 - clove::ImageView, 540
 - clove::Label, 561
 - clove::ListView, 594
 - clove::MainView, 620
 - clove::MediaPlayer, 646
 - clove::Menubar, 668
 - clove::ModalPanel, 688
 - clove::MultilineEditBox, 712
 - clove::NumericEditBox, 756
 - clove::PasswordEditBox, 780
 - clove::PopupMenu, 801
 - clove::PopupMenuButton, 824
 - clove::ProgressBar, 845
 - clove::RadioButton, 874
 - clove::RawResizeSplitter, 897
- setVisibility
 - clove::AbstractMenu, 42
 - clove::AudioPlayer, 86
 - clove::Border, 107
 - clove::Button, 129
 - clove::Carousel, 153
 - clove::CheckButton, 175
 - clove::DataView, 226
 - clove::DateBox, 251
 - clove::Dialog, 272
 - clove::DropDownBox, 296
 - clove::EditBox, 320
 - clove::EditComboBox, 343
 - clove::Expander, 369
 - clove::FlatView, 406
 - clove::Form, 426
 - clove::Grid, 448
 - clove::HorizontalStack, 477
 - clove::HtmlView, 496
 - clove::IconView, 520
 - clove::ImageView, 540
 - clove::Label, 561
 - clove::ListView, 594
 - clove::MainView, 620
 - clove::MediaPlayer, 647
 - clove::Menubar, 668
 - clove::ModalPanel, 689
 - clove::MultilineEditBox, 712
 - clove::NumericEditBox, 756
 - clove::PasswordEditBox, 780
 - clove::PopupMenu, 801
 - clove::PopupMenuButton, 824
 - clove::ProgressBar, 845
 - clove::RadioButton, 874
 - clove::RawResizeSplitter, 897

- clove::ResizeSplitter, 921
- clove::RichEdit, 941
- clove::RichEditBox, 963
- clove::ScrollView, 987
- clove::Slider, 1009
- clove::Spacer, 1038
- clove::TabView, 1096
- clove::TableView, 1072
- clove::TimeBox, 1120
- clove::ToolBar, 1142
- clove::TreeView, 1173
- clove::VerticalStack, 1208
- clove::VideoPlayer, 1236
- clove::Widget, 1256
- clove::Wrap, 1276
- setVolume
 - clove::AudioPlayer, 87
 - clove::MediaPlayer, 647
 - clove::VideoPlayer, 1236
- setVstretch
 - clove::AbstractMenu, 42
 - clove::AudioPlayer, 87
 - clove::Border, 108
 - clove::Button, 129
 - clove::Carousel, 154
 - clove::CheckBox, 175
 - clove::DataView, 226
 - clove::DateBox, 251
 - clove::Dialog, 272
 - clove::DropDownBox, 297
 - clove::EditBox, 320
 - clove::EditComboBox, 344
 - clove::Expander, 369
 - clove::FlatView, 406
 - clove::Form, 427
 - clove::Grid, 448
 - clove::HorizontalStack, 477
 - clove::HtmlView, 497
 - clove::IconView, 520
 - clove::ImageView, 540
 - clove::Label, 562
 - clove::ListView, 594
 - clove::MainView, 621
 - clove::MediaPlayer, 647
 - clove::Menubar, 668
 - clove::ModalPanel, 689
 - clove::MultilineEditBox, 712
 - clove::NumericEditBox, 756
 - clove::PasswordEditBox, 780
 - clove::PopupMenu, 801
 - clove::PopupMenuButton, 824
 - clove::ProgressBar, 845
 - clove::RadioButton, 875
 - clove::RawResizeSplitter, 898
 - clove::ResizeSplitter, 921
 - clove::RichEdit, 942
 - clove::RichEditBox, 963
 - clove::ScrollView, 987
 - clove::Slider, 1009
 - clove::Spacer, 1038
 - clove::TabView, 1096
 - clove::TableView, 1072
 - clove::TimeBox, 1120
 - clove::ToolBar, 1143
 - clove::TreeView, 1173
 - clove::VerticalStack, 1209
 - clove::VideoPlayer, 1236
 - clove::Widget, 1256
 - clove::Wrap, 1277
- setZoom
 - clove::ImageView, 541
- show
 - clove::Dialog, 272
 - clove::PopupMenu, 802
- showChangeMenu
 - clove::DataView, 227
 - clove::ListView, 595
 - clove::TableView, 1073
 - clove::TreeView, 1174
- showControls
 - clove::AudioPlayer, 87
 - clove::MediaPlayer, 647
 - clove::VideoPlayer, 1236
- showOnly
 - clove::MainView, 621
 - clove::ResizeSplitter, 922
- showOnlyFirstColumn
 - clove::DataView, 227
 - clove::ListView, 595
 - clove::TableView, 1073
 - clove::TreeView, 1174
- sibling
 - clove::DatasourceValuePointer, 196
- sidebar
 - clove::MainView, 621
- size
 - clove::IconView, 520
- sizeFractions
 - clove::ResizeSplitter, 922
- skipExecution
 - clove::EventArgs, 349
- SortProxyDatasource
 - clove::SortProxyDatasource, 1013
- source
 - clove::AudioPlayer, 87
 - clove::ImageView, 541
 - clove::MediaPlayer, 648
 - clove::VideoPlayer, 1237
- splitterPosition
 - clove::MainView, 621
- splitterVisibility
 - clove::ResizeSplitter, 922
- splitterWidth
 - clove::ResizeSplitter, 922
- stepsize
 - clove::NumericEditBox, 757

- clove::Slider, [1010](#)
 - strictHorizontalSizing
 - clove::AbstractMenu, [42](#)
 - clove::AudioPlayer, [88](#)
 - clove::Border, [108](#)
 - clove::Button, [129](#)
 - clove::Carousel, [154](#)
 - clove::CheckBox, [176](#)
 - clove::DataView, [227](#)
 - clove::DateBox, [252](#)
 - clove::Dialog, [273](#)
 - clove::DropDownBox, [297](#)
 - clove::EditBox, [320](#)
 - clove::EditComboBox, [344](#)
 - clove::Expander, [369](#)
 - clove::FlatView, [407](#)
 - clove::Form, [427](#)
 - clove::Grid, [448](#)
 - clove::HorizontalStack, [477](#)
 - clove::HtmlView, [497](#)
 - clove::IconView, [520](#)
 - clove::ImageView, [541](#)
 - clove::Label, [562](#)
 - clove::ListView, [595](#)
 - clove::MainView, [621](#)
 - clove::MediaPlayer, [648](#)
 - clove::Menubar, [668](#)
 - clove::ModalPanel, [689](#)
 - clove::MultilineEditBox, [712](#)
 - clove::NumericEditBox, [757](#)
 - clove::PasswordEditBox, [780](#)
 - clove::PopupMenu, [802](#)
 - clove::PopupMenuButton, [824](#)
 - clove::ProgressBar, [846](#)
 - clove::RadioButton, [875](#)
 - clove::RawResizeSplitter, [898](#)
 - clove::ResizeSplitter, [922](#)
 - clove::RichEdit, [942](#)
 - clove::RichEditBox, [964](#)
 - clove::ScrollView, [987](#)
 - clove::Slider, [1010](#)
 - clove::Spacer, [1039](#)
 - clove::TabView, [1096](#)
 - clove::TableView, [1073](#)
 - clove::TimeBox, [1121](#)
 - clove::ToolBar, [1143](#)
 - clove::TreeView, [1174](#)
 - clove::VerticalStack, [1209](#)
 - clove::VideoPlayer, [1237](#)
 - clove::Widget, [1257](#)
 - clove::Wrap, [1277](#)
 - strictVerticalSizing
 - clove::AbstractMenu, [43](#)
 - clove::AudioPlayer, [88](#)
 - clove::Border, [108](#)
 - clove::Button, [129](#)
 - clove::Carousel, [154](#)
 - clove::CheckBox, [176](#)
 - clove::DataView, [227](#)
 - clove::DateBox, [252](#)
 - clove::Dialog, [273](#)
 - clove::DropDownBox, [297](#)
 - clove::EditBox, [320](#)
 - clove::EditComboBox, [344](#)
 - clove::Expander, [369](#)
 - clove::FlatView, [407](#)
 - clove::Form, [427](#)
 - clove::Grid, [449](#)
 - clove::HorizontalStack, [477](#)
 - clove::HtmlView, [497](#)
 - clove::IconView, [520](#)
 - clove::ImageView, [541](#)
 - clove::Label, [562](#)
 - clove::ListView, [595](#)
 - clove::MainView, [622](#)
 - clove::MediaPlayer, [648](#)
 - clove::Menubar, [669](#)
 - clove::ModalPanel, [689](#)
 - clove::MultilineEditBox, [712](#)
 - clove::NumericEditBox, [757](#)
 - clove::PasswordEditBox, [780](#)
 - clove::PopupMenu, [802](#)
 - clove::PopupMenuButton, [825](#)
 - clove::ProgressBar, [846](#)
 - clove::RadioButton, [875](#)
 - clove::RawResizeSplitter, [898](#)
 - clove::ResizeSplitter, [922](#)
 - clove::RichEdit, [942](#)
 - clove::RichEditBox, [964](#)
 - clove::ScrollView, [988](#)
 - clove::Slider, [1010](#)
 - clove::Spacer, [1039](#)
 - clove::TabView, [1096](#)
 - clove::TableView, [1073](#)
 - clove::TimeBox, [1121](#)
 - clove::ToolBar, [1143](#)
 - clove::TreeView, [1174](#)
 - clove::VerticalStack, [1209](#)
 - clove::VideoPlayer, [1237](#)
 - clove::Widget, [1257](#)
 - clove::Wrap, [1277](#)
- style
- clove::AbstractMenu, [43](#)
 - clove::AudioPlayer, [88](#)
 - clove::Border, [108](#)
 - clove::Button, [130](#)
 - clove::Carousel, [154](#)
 - clove::CheckBox, [176](#)
 - clove::DataView, [228](#)
 - clove::DateBox, [252](#)
 - clove::Dialog, [273](#)
 - clove::DropDownBox, [297](#)
 - clove::EditBox, [321](#)
 - clove::EditComboBox, [344](#)
 - clove::Expander, [370](#)
 - clove::FlatView, [407](#)

- clove::Form, [427](#)
- clove::Grid, [449](#)
- clove::HorizontalStack, [477](#)
- clove::HtmlView, [497](#)
- clove::IconView, [521](#)
- clove::ImageView, [541](#)
- clove::Label, [562](#)
- clove::ListView, [596](#)
- clove::MainView, [622](#)
- clove::MediaPlayer, [648](#)
- clove::Menubar, [669](#)
- clove::ModalPanel, [690](#)
- clove::MultilineEditBox, [713](#)
- clove::NumericEditBox, [757](#)
- clove::PasswordEditBox, [781](#)
- clove::PopupMenu, [802](#)
- clove::PopupMenuButton, [825](#)
- clove::ProgressBar, [846](#)
- clove::RadioButton, [875](#)
- clove::RawResizeSplitter, [898](#)
- clove::ResizeSplitter, [923](#)
- clove::RichEdit, [942](#)
- clove::RichEditBox, [964](#)
- clove::ScrollView, [988](#)
- clove::Slider, [1010](#)
- clove::Spacer, [1039](#)
- clove::TabView, [1097](#)
- clove::TableView, [1074](#)
- clove::TimeBox, [1121](#)
- clove::Toolbar, [1143](#)
- clove::TreeView, [1175](#)
- clove::VerticalStack, [1209](#)
- clove::VideoPlayer, [1237](#)
- clove::Widget, [1257](#)
- clove::Wrap, [1277](#)
- styleClass
 - clove::AbstractMenu, [43](#)
 - clove::AudioPlayer, [88](#)
 - clove::Border, [108](#)
 - clove::Button, [130](#)
 - clove::Carousel, [154](#)
 - clove::CheckButton, [176](#)
 - clove::DataView, [228](#)
 - clove::DateBox, [252](#)
 - clove::Dialog, [273](#)
 - clove::DropDownBox, [297](#)
 - clove::EditBox, [321](#)
 - clove::EditComboBox, [344](#)
 - clove::Expander, [370](#)
 - clove::FlatView, [407](#)
 - clove::Form, [427](#)
 - clove::Grid, [449](#)
 - clove::HorizontalStack, [478](#)
 - clove::HtmlView, [497](#)
 - clove::IconView, [521](#)
 - clove::ImageView, [542](#)
 - clove::Label, [562](#)
 - clove::ListView, [596](#)
 - clove::MainView, [622](#)
 - clove::MediaPlayer, [648](#)
 - clove::Menubar, [669](#)
 - clove::ModalPanel, [690](#)
 - clove::MultilineEditBox, [713](#)
 - clove::NumericEditBox, [757](#)
 - clove::PasswordEditBox, [781](#)
 - clove::PopupMenu, [803](#)
 - clove::PopupMenuButton, [825](#)
 - clove::ProgressBar, [846](#)
 - clove::RadioButton, [875](#)
 - clove::RawResizeSplitter, [898](#)
 - clove::ResizeSplitter, [923](#)
 - clove::RichEdit, [942](#)
 - clove::RichEditBox, [964](#)
 - clove::ScrollView, [988](#)
 - clove::Slider, [1010](#)
 - clove::Spacer, [1039](#)
 - clove::TabView, [1097](#)
 - clove::TableView, [1074](#)
 - clove::TimeBox, [1121](#)
 - clove::Toolbar, [1143](#)
 - clove::TreeView, [1175](#)
 - clove::VerticalStack, [1209](#)
 - clove::VideoPlayer, [1237](#)
 - clove::Widget, [1257](#)
 - clove::Wrap, [1277](#)
- suspendResizing
 - clove::utils, [1186](#)
- switchInvisibleAnimationDuration
 - clove::Carousel, [155](#)
 - clove::FlatLayout, [384](#)
 - clove::FlatView, [407](#)
 - clove::TabView, [1097](#)
- switchInvisibleAnimationName
 - clove::Carousel, [155](#)
 - clove::FlatLayout, [384](#)
 - clove::FlatView, [407](#)
 - clove::TabView, [1097](#)
- switchToLeftBody
 - clove::MainView, [622](#)
- switchToRightBody
 - clove::MainView, [622](#)
- switchVisibleAnimationDuration
 - clove::Carousel, [155](#)
 - clove::FlatLayout, [385](#)
 - clove::FlatView, [408](#)
 - clove::TabView, [1097](#)
- switchVisibleAnimationName
 - clove::Carousel, [155](#)
 - clove::FlatLayout, [385](#)
 - clove::FlatView, [408](#)
 - clove::TabView, [1097](#)
- tabBarLocation
 - clove::Carousel, [155](#)
 - clove::TabView, [1098](#)
- tabs
 - clove::Carousel, [155](#)

- clove::TabView, [1098](#)
- text
 - clove::DateBox, [252](#)
 - clove::DropDownBox, [298](#)
 - clove::EditBox, [321](#)
 - clove::EditComboBox, [345](#)
 - clove::MultilineEditBox, [713](#)
 - clove::NumericEditBox, [758](#)
 - clove::PasswordEditBox, [781](#)
 - clove::TimeBox, [1121](#)
- textSelection
 - clove::DateBox, [252](#)
 - clove::DropDownBox, [298](#)
 - clove::EditBox, [321](#)
 - clove::EditComboBox, [345](#)
 - clove::MultilineEditBox, [713](#)
 - clove::NumericEditBox, [758](#)
 - clove::PasswordEditBox, [781](#)
 - clove::TimeBox, [1121](#)
- time
 - clove::TimeBox, [1122](#)
- title
 - clove::Dialog, [273](#)
- titleicon
 - clove::Dialog, [274](#)
- toColumn
 - clove::NativeDatasourceNode, [732](#)
- toPath
 - clove::DatasourceValuePointer, [197](#)
- toRow
 - clove::NativeDatasourceNode, [732](#)
- toString
 - clove::DatasourceValuePointer, [197](#)
- toggleSelectionBold
 - clove::RichEditBox, [964](#)
- toggleSelectionItalic
 - clove::RichEditBox, [964](#)
- toggleSelectionUnderline
 - clove::RichEditBox, [965](#)
- trigger
 - clove::Event, [348](#)
- tryLoadScript
 - clove::utils, [1186](#)
- tryLoadStyle
 - clove::utils, [1187](#)
- unbind
 - clove::DataBinding, [188](#)
- unregisterBusy
 - clove::AbstractMenu, [43](#)
 - clove::AudioPlayer, [88](#)
 - clove::Border, [109](#)
 - clove::Button, [130](#)
 - clove::Carousel, [155](#)
 - clove::CheckButton, [176](#)
 - clove::DataView, [228](#)
 - clove::DateBox, [252](#)
 - clove::Dialog, [274](#)
 - clove::DropDownBox, [298](#)
 - clove::EditBox, [321](#)
 - clove::EditComboBox, [345](#)
 - clove::Expander, [370](#)
 - clove::FlatView, [408](#)
 - clove::Form, [428](#)
 - clove::Grid, [449](#)
 - clove::HorizontalStack, [478](#)
 - clove::HtmlView, [498](#)
 - clove::IconView, [521](#)
 - clove::ImageView, [542](#)
 - clove::Label, [563](#)
 - clove::ListView, [596](#)
 - clove::MainView, [622](#)
 - clove::MediaPlayer, [648](#)
 - clove::Menubar, [669](#)
 - clove::ModalPanel, [690](#)
 - clove::MultilineEditBox, [713](#)
 - clove::NumericEditBox, [758](#)
 - clove::PasswordEditBox, [781](#)
 - clove::PopupMenu, [803](#)
 - clove::PopupMenuButton, [825](#)
 - clove::ProgressBar, [846](#)
 - clove::RadioButton, [876](#)
 - clove::RawResizeSplitter, [899](#)
 - clove::ResizeSplitter, [923](#)
 - clove::RichEdit, [943](#)
 - clove::RichEditBox, [965](#)
 - clove::ScrollView, [988](#)
 - clove::Slider, [1010](#)
 - clove::Spacer, [1039](#)
 - clove::TabView, [1098](#)
 - clove::TableView, [1074](#)
 - clove::TimeBox, [1122](#)
 - clove::ToolBar, [1144](#)
 - clove::TreeView, [1175](#)
 - clove::VerticalStack, [1210](#)
 - clove::VideoPlayer, [1237](#)
 - clove::Widget, [1257](#)
 - clove::Wrap, [1278](#)
- unselectAll
 - clove::RadioGroup, [879](#)
- userMayAddTabs
 - clove::Carousel, [156](#)
 - clove::TabView, [1098](#)
- value
 - clove::NumericEditBox, [758](#)
 - clove::ProgressBar, [847](#)
 - clove::Slider, [1011](#)
- valueDef
 - clove::RadioButton, [876](#)
- valuePointer
 - clove::AjaxAsyncDatasource, [54](#)
 - clove::AsyncDatasource, [62](#)
 - clove::Datasource, [192](#)
 - clove::FilterProxyDatasource, [379](#)
 - clove::NativeDatasource, [725](#)
 - clove::NativeDatasourceNode, [732](#)
 - clove::ProxyDatasource, [854](#)

- clove::SortProxyDatasource, 1020
- valuePointerNavigateInDepth
 - clove::AjaxAsyncDatasource, 54
 - clove::AsyncDatasource, 63
 - clove::Datasource, 193
 - clove::FilterProxyDatasource, 380
 - clove::NativeDatasource, 726
 - clove::ProxyDatasource, 854
 - clove::SortProxyDatasource, 1020
- valuePointerToNativeNode
 - clove::NativeDatasource, 726
- verticalScrollPosition
 - clove::ScrollView, 988
- verticalStretchAffinity
 - clove::AbstractMenu, 43
 - clove::AudioPlayer, 89
 - clove::Border, 109
 - clove::Button, 130
 - clove::Carousel, 156
 - clove::CheckBox, 177
 - clove::DataView, 228
 - clove::DateBox, 253
 - clove::Dialog, 274
 - clove::DropDownBox, 298
 - clove::EditBox, 322
 - clove::EditComboBox, 345
 - clove::Expander, 370
 - clove::FlatView, 408
 - clove::Form, 428
 - clove::Grid, 449
 - clove::HorizontalStack, 478
 - clove::HtmlView, 498
 - clove::IconView, 521
 - clove::ImageView, 542
 - clove::Label, 563
 - clove::ListView, 596
 - clove::MainView, 623
 - clove::MediaPlayer, 649
 - clove::Menubar, 669
 - clove::ModalPanel, 690
 - clove::MultilineEditBox, 714
 - clove::NumericEditBox, 758
 - clove::PasswordEditBox, 782
 - clove::PopupMenu, 803
 - clove::PopupMenuButton, 825
 - clove::ProgressBar, 847
 - clove::RadioButton, 876
 - clove::RawResizeSplitter, 899
 - clove::ResizeSplitter, 923
 - clove::RichEdit, 943
 - clove::RichEditBox, 965
 - clove::ScrollView, 989
 - clove::Slider, 1011
 - clove::Spacer, 1040
 - clove::TabView, 1098
 - clove::TableView, 1074
 - clove::TimeBox, 1122
 - clove::Toolbar, 1144
 - clove::TreeView, 1175
 - clove::VerticalStack, 1210
 - clove::VideoPlayer, 1238
 - clove::Widget, 1258
 - clove::Wrap, 1278
- visibility
 - clove::AbstractMenu, 44
 - clove::AudioPlayer, 89
 - clove::Border, 109
 - clove::Button, 130
 - clove::Carousel, 156
 - clove::CheckBox, 177
 - clove::DataView, 228
 - clove::DateBox, 253
 - clove::Dialog, 274
 - clove::DropDownBox, 298
 - clove::EditBox, 322
 - clove::EditComboBox, 345
 - clove::Expander, 370
 - clove::FlatView, 408
 - clove::Form, 428
 - clove::Grid, 450
 - clove::HorizontalStack, 478
 - clove::HtmlView, 498
 - clove::IconView, 521
 - clove::ImageView, 542
 - clove::Label, 563
 - clove::ListView, 596
 - clove::MainView, 623
 - clove::MediaPlayer, 649
 - clove::Menubar, 670
 - clove::ModalPanel, 690
 - clove::MultilineEditBox, 714
 - clove::NumericEditBox, 759
 - clove::PasswordEditBox, 782
 - clove::PopupMenu, 803
 - clove::PopupMenuButton, 826
 - clove::ProgressBar, 847
 - clove::RadioButton, 876
 - clove::RawResizeSplitter, 899
 - clove::ResizeSplitter, 923
 - clove::RichEdit, 943
 - clove::RichEditBox, 965
 - clove::ScrollView, 989
 - clove::Slider, 1011
 - clove::Spacer, 1040
 - clove::TabView, 1099
 - clove::TableView, 1074
 - clove::TimeBox, 1122
 - clove::Toolbar, 1144
 - clove::TreeView, 1175
 - clove::VerticalStack, 1210
 - clove::VideoPlayer, 1238
 - clove::Widget, 1258
 - clove::Wrap, 1278
- Visible
 - clove, 186
- volume

- clove::AudioPlayer, [89](#)
- clove::MediaPlayer, [649](#)
- clove::VideoPlayer, [1238](#)
- vstretch
 - clove::AbstractMenu, [44](#)
 - clove::AudioPlayer, [89](#)
 - clove::Border, [109](#)
 - clove::Button, [131](#)
 - clove::Carousel, [156](#)
 - clove::CheckBox, [177](#)
 - clove::DataView, [229](#)
 - clove::DateBox, [253](#)
 - clove::Dialog, [274](#)
 - clove::DropDownBox, [299](#)
 - clove::EditBox, [322](#)
 - clove::EditComboBox, [346](#)
 - clove::Expander, [371](#)
 - clove::FlatView, [409](#)
 - clove::Form, [428](#)
 - clove::Grid, [450](#)
 - clove::HorizontalStack, [478](#)
 - clove::HtmlView, [498](#)
 - clove::IconView, [522](#)
 - clove::ImageView, [542](#)
 - clove::Label, [563](#)
 - clove::ListView, [597](#)
 - clove::MainView, [623](#)
 - clove::MediaPlayer, [649](#)
 - clove::Menubar, [670](#)
 - clove::ModalPanel, [691](#)
 - clove::MultilineEditBox, [714](#)
 - clove::NumericEditBox, [759](#)
 - clove::PasswordEditBox, [782](#)
 - clove::PopupMenu, [803](#)
 - clove::PopupMenuButton, [826](#)
 - clove::ProgressBar, [847](#)
 - clove::RadioButton, [876](#)
 - clove::RawResizeSplitter, [899](#)
 - clove::ResizeSplitter, [924](#)
 - clove::RichEdit, [943](#)
 - clove::RichEditBox, [965](#)
 - clove::ScrollView, [989](#)
 - clove::Slider, [1011](#)
 - clove::Spacer, [1040](#)
 - clove::TabView, [1099](#)
 - clove::TableView, [1075](#)
 - clove::TimeBox, [1122](#)
 - clove::Toolbar, [1144](#)
 - clove::TreeView, [1176](#)
 - clove::VerticalStack, [1210](#)
 - clove::VideoPlayer, [1238](#)
 - clove::Widget, [1258](#)
 - clove::Wrap, [1278](#)
- Widget
 - clove::Widget, [1242](#)
- zoom
 - clove::ImageView, [543](#)