Contents

Chapter 1

Shallot Scripting Documentation

1.1 Scripting Manual

Shallot has a plugin interface which allows to add functionality to the core from outside. This document is for people who are going to create a new Shallot plugin in order to bring some new functionality to Shallot.

A Shallot plugin developer should have at least basic skills in the `Python` programming language and should have read the entire section Scripting Manual (From the Feature Overview, it might be okay to read just the parts relevant for you).

After the Scripting Manual, you find the complete interface reference. You should use it for finding some technical details (which classes exist, which methods are there, which arguments do they have, ...). It lists the structure of the interface from different perspectives. Some of them are quite useful, others ones are more technical and only interesting in rare situations. Good starting points are those sections:

- The "Class List" provides an overview of all existing classes.
- The "shallot Namespace Reference" gives a coarser overview of all existing subparts.

1.1.1 General Notes

This section covers some notes and hints, which you should know about before you begin with developing Shallot plugins.

1.1.1.1 Plugin Management Assistant

Shallot includes a plugin manager. It helps to work with plugins as it provides an overview of all installed plugins and some management actions you can execute on them (e.g. deleting, disabling).

It is not enabled by default, since it is not required for day-to-day use. For plugin development, you should enable the Plugin Management in the *Tuning* dialog.

It also provides a collection of sample plugins. They can either be used as some kind of a tiny step-by-step tutorial or as templates for your new plugins.

1.1.1.2 General Plugin Design

Shallot plugins are Python scripts in a single file with the filename `[pluginname].py` (replacing [pluginname] with the actual name). They must `import shallot` and use the methods provided there for interacting with the Shallot core.

1.1.1.3 Implement & Register

For the most kinds of functionality you can add to Shallot, the development pattern is as follows (everything happening in your Python plugin file):

- At first you subclass one of the Shallot base classes and put your functionality into the class implementation. The documentation of the base class tells which methods may and/or must be implemented in your subclass.
- Then, in the *main* section of that plugin - this is the unindented root block as it is in typical executable Python scripts - the code must add this new subclass to the Shallot runtime. This happens with one of the `register*` methods in the Shallot interface.

Which particular elements of the Shallot interface are used, depends on what kind of functionality you want to add. The section Feature Overview below describes the details.

1.1.1.4 Position Indexes

For some kinds of pluggable functionality, Shallot keeps an *ordered* list of plugged objects. The semantic of the order can differ, e.g. it can be used for displaying purposes or for an execution order.

The technical mechanism is similar on most places in the interface. Particular functions (i.e. some constructors and register methods) have this two arguments amongst others:

- `positionGroup`: A coarse order information. This must be set to one of the elements in a particular enumeration (see the function documentation for details).
- `positionIndex`: An integer between 0 and (at least) 10000 which controls the order within the chosen group. Lower values mean sooner placement in the order. They are not required to be continuous but can have holes and also duplicates (although the order of duplicates to each other is then undefined). Existing larger indexes will also not shift (as they would do in a typical list) on insertion of a new item.

The documentation texts of them explain what the order is used for. Those two arguments together control the placement in the ordered object list. Typically both ones are optional and have a default:

- `positionGroup`: Some fixed default group. The function documentation might mention details.
- `positionIndex`: Some random-like integer, which is derived from your class name (and so is always the same in each run on each machine).

If so, you can decide to use those defaults, or to manually set either one or both of them.

If the order is critical (e.g. because it is the execution order of a process which only succeeds in a certain order), it is probably required to manually set them. Obviously you must then find out the values from the related objects (e.g. by trial-and-error) and tune your values according to them.

If not, you are fine with the defaults in many cases.

1.1.1.5 The '_ApiProxy' Class

Studying the reference, you will see the class `shallot._ApiProxy` be referred at many places. Whenever this is the case, e.g. as a superclass of some interface classes, this is mostly a technical information with no big impact on the developer. It is a common superclass for many classes in the Shallot scripting interface, since it gives them access to some internal parts of the infrastructure. You should not directly come in touch with any specifics of this class.

1.1.1.6 Exceptions

The base class for exceptions in Shallot is shallot.Exceptions.ScriptedException. The shallot.Exceptions namespace contains more interesting stuff.

Try to only throw those exceptions (either directly or indirectly) from your code to be failsafe. You should use one of the subclasses in `raise` statements. You should never use subclasses in `expect [Foo]` statements but only shallot.Exceptions.ScriptedException. Use shallot.Exceptions.ScriptedException.isExceptionClass for checking against a certain class (as string) and rethrow if needed. This is because a exception in Shallot typically has more classes than the direct Python side hierarchy would look like.

1.1.1.7 Localization

Shallot plugins can support more than just one language. For multi-language support, you should create a global instance of shallot.IntlStringMap at the beginning and take strings from there when they are used for displaying:

```
import shallot

Strings = shallot.IntlStringMap(
    HelloWorld = {"en":"Hello world!", "de":"Hallo Welt!"},
    SomethingElse = {"en":"Something different", "de":"Etwas anderes"},
)

shallot.Logging.log_info(Strings.HelloWorld)
```

1.1.1.8 Deployment

The deployment of a Shallot plugin is as easy as copying the plugin file. However, the destination differs among the various operating systems. Since there is a system-global location and a per-user location, there is one more degree of variation.

On the target system, you can find out the actual destination paths by means of the 'Find in filesystem' action in the Plugin Manager.

Once the destination path is known, you can choose an arbitrary deployment technique - e.g. manual file copying, or using the native package manager from the operating system - for bringing the plugin file in place.

1.1.2 Feature Overview

This sections introduces the different kinds of functionality you can add by means of this plugin interface. Those are the technical building blocks you have to use for implementing your plugin logic.

Most of the sections begin with an abstract explanation of the mechanisms. Read ahead! It should become clear later in the text. Each section includes links to the elements in the scripting interface reference which you need to know. You should read the documentations of this elements.

1.1.2.1 Actions

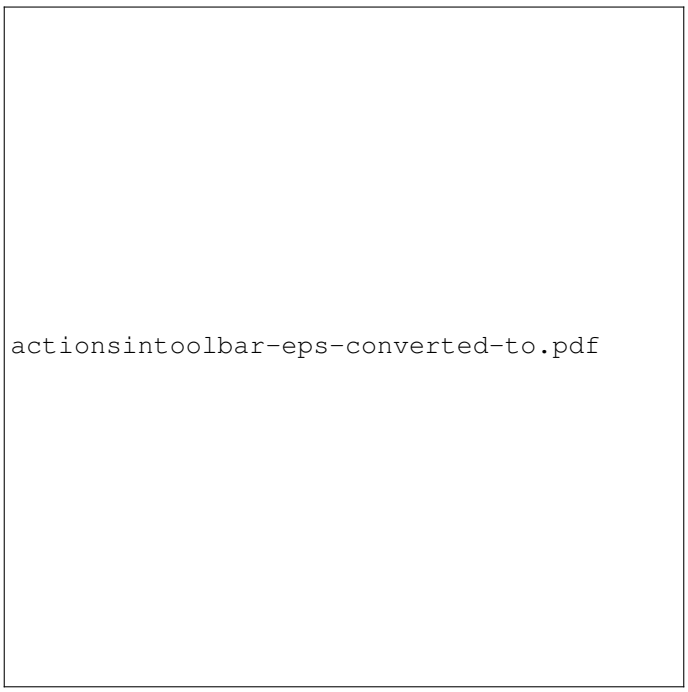
As a first approximation, an action is some piece of code for execution. Conceptually an action is similar to a simple Python function (although it technically is not just a function as you will read later). However, executing them brings a lot of additional functionality (e.g. dialog support for user communication) and infrastructure.

Actions are used in different ways in Shallot:

- The easiest thing to do with an action is to simply execute it. This is similar to directly executing the code, but using the additional functionality.
- Another common usage is to offer them in the user interface for some files or directories, so the user can decide to execute them.
- There are some other places in the plugin interface, which work with actions somehow. Read those documentations for details.

The first way gives a very versatile tool. In most places in the plugin code, you can put some parts of your routine into an action and execute it in order to use the additional functionality we will see later.

The second way enables your plugin to provide some code for a given selection of files and/or directories for the user, so the user can manually trigger it. Those are typically represented in context menus or the toolbar.



At this point, our first approximation must see some corrections:

An action is a (indirect) subclass of shallot.Actions.AbstractAction. It has the following subclasses, which are typical base classes for custom implementations:

- shallot.Actions.ActionAction: This kind of action is executable.
- shallot.Actions.SubmenuAction: A submenu is a list of other actions. It is not directly executable, but can be presented as a submenu to the user. This allows to bundle multiple shallot.Actions.ActionAction implementations which provide related functionality for better overview.

A shallot.Actions.ActionAction instance (i.e. an instance of your subclass) can be executed by initializing it at first (calling shallot.Actions.ActionAction.initialize_sync) and calling shallot.Actions.ActionAction.execute.

For sample code, read the sources of the 'action' sample plugin (in the Plugin Management).

In order to provide your action subclass in the user interface, call shallot.Actions.register.

For sample code, read the sources of the 'action.advanced' sample plugin.

The additional functionality comes with the `info` parameter you get in your shallot.Actions.ActionAction.action implementation. It is an instance of shallot.Actions.ExecutionInfo. Read this documentation for a full overview. One essential thing you can do with such an object is to have dialogs with the user. The next section shows how to do this.

The shallot.Actions namespace contains more interesting stuff.

1.1.2.2 User Feedback

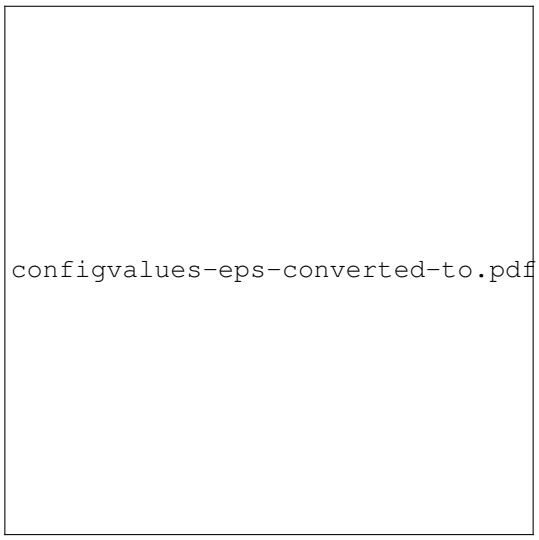
User feedback is everything about giving some information to the user and waiting for its answer in an interactive way.

Typical situations where user feedback operations take place are Action executions. There an instance of shallot.↩ Actions.ExecutionInfo is available. shallot.Actions.ExecutionInfo.userfeedback instance gives you access to an instance of shallot.Actions.ExecutionUserFeedback, which is the key element to all kinds of feedback operations.

For sample code, read the sources of the 'userfeedback' sample plugin (in the Plugin Management).

1.1.2.3 Configuration Values

A plugin can use own configuration values which are stored by Shallot and are visible to the user in the *Tuning* dialog.



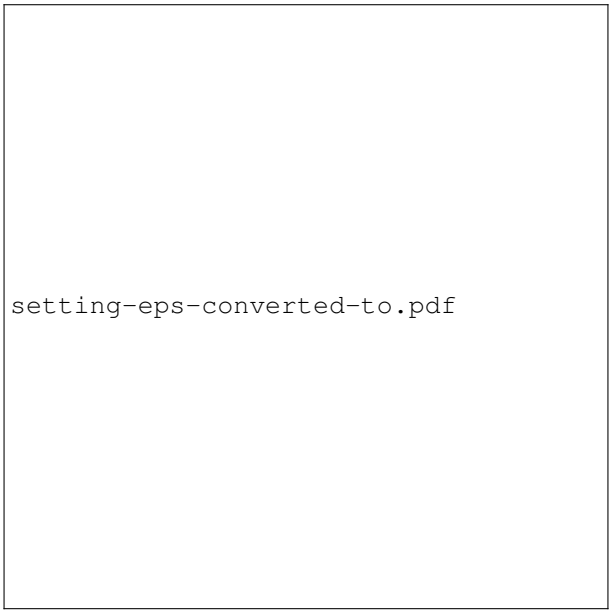
They are fine for making stuff configurable for special situations, which should typically should stay at default. For many other situations, e.g. when the user is expected to changes this value regularly, check if the Settings features fit better.

Using an own configuration value begins with subclassing shallot.ConfigurationValue. Afterwards, you should create one instance of this class globally and use its methods. No registration is required.

For sample code, read the sources of the 'configurationvalue' sample plugin (in the Plugin Management).

1.1.2.4 Settings

Settings are a common mechanism for various kinds of customizations the user can make in its Shallot environment. The user can store them, bind them to particular subdirectories and profiles and manage them in the *Settings* dialog.



A plugin can participate in this mechanism by adding own settings.

This begins with subclassing shallot.Setting. Afterwards, shallot.Setting.register must be used for registering it.

For sample code, read the sources of the 'setting' sample plugin (in the Plugin Management).

1.1.2.5 Main Window

The shallot.MainWindow class provides some navigation function and more. Read the class documentation for details. The main window object is globally available as shallot.MainWindow.current.

For sample code, read the sources of the 'mainwindow' sample plugin (in the Plugin Management).

1.1.2.6 EURLs

EURLs are addresses of objects in the filesystem. They are a very basic part of the Shallot foundation.

Conceptually EURLs are similar to `URLs`. In order to express cascades of locations, EURLs are defined as a superset of URLs. As a meaningful example, this EURL refers to a file in an archive file, which in turn is located on a remote server:

```
zip:/[ftp://ftpserver/foo/bar.zip]//zippedfolder/zippedfile
```

This cascades can have arbitrary depths (although this obviously doesn't make sense in arbitrary combinations).

EURLs are used in many different parts in the scripting interface. They are instances of the class shallot.Eurl, which provide a rich set of getters and creator functions.

For sample code, read the sources of the 'eurl' sample plugin (in the Plugin Management).

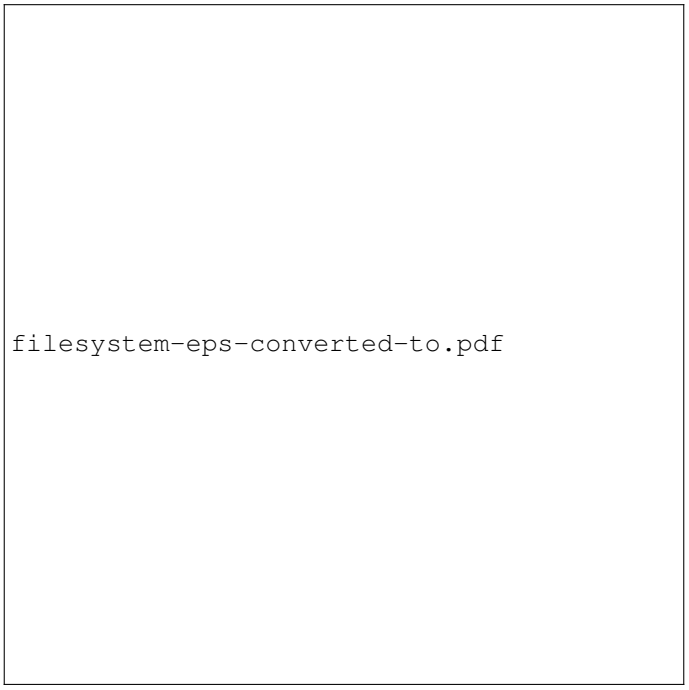
1.1.2.7 Filesystem

Shallot keeps an own model of the entire filesystem tree in memory (technically, it determines and stores stuff just on demand, of course). A Shallot plugin has different ways to work with this model.

At first, there are some methods for requesting some informations directly from the model. Find the static methods in shallot.Filesystem for more details (not all of them request things but some are relevant for the stuff explained later). A key class in interactions with the model, also used at various different parts in the scripting interface, is shallot.Filesystem.Node.

For sample code, read the sources of the 'filesystemnode' sample plugin (in the Plugin Management).

A different way of using it filesystem model is to provide custom implementations and own nodes in the tree.



This at least requires to subclass shallot.Filesystem.Handler and registering it with shallot.Filesystem.Handler.↩ register.

In typical scenarios it also requires to create a shallot.Filesystem.Node which is associated with this handler and to insert it into the model. Some of the static methods of shallot.Filesystem are used for this procedure. Please note that it is just required to insert root nodes somewhere. The subtree of that node is specified by your handler implementation, but the node insertions (and removals) take place implicitely.

You should also read about Operations, since you might have to deal with the key object of this feature here as well.

For sample code, read the sources of the 'filesystem' sample plugin (in the Plugin Management).

A filesystem handler has a wide range of possibilities for extending the Shallot functionality. One of them are custom detail columns, which are subject of the next section.

1.1.2.8 Detail Columns

A custom detail column, as it is visible in file views, can offer additional information to the user.

Implementing custom detail columns begins with subclassing shallot.DetailColumn. You should then create one instance of this class and add it to some nodes with shallot.Filesystem.Node.add_detail.

For sample code, read the sources of the 'filesystem' sample plugin (in the Plugin Management).

1.1.2.9 Operations

Operations are a concept of bundling a bunch of small changes in the filesystem together to one large step. They are also essential for working in cascaded filesystems as they are expressed with EURLs. Operations are entirely a backend concept without any direct representations in the user interface.

A small example explains why this bundling is a required thing: There is a ftp server which contains two archive files; `a.zip` (A) and `b.zip` (B). They build the roots of the subtrees `zip:/[ftp://foo/a.zip]//` and `zip↩ :/[ftp://foo/b.zip]//`, each containing the contents of the respective archive. The user requests a large file copy action for 1000 files in `zip:/[ftp://foo/a.zip]//` to somewhere in `zip:/[ftp://foo/b.↩ zip]//`. In an unbundled way, Shallot would at first fetch A, then extract the first file, then fetch B, put the file content into the archive and upload the new version of B again. Afterwards, it would run the same loop for the second file, then for the third, ... In an operation, Shallot would fetch A and B just once, then works on the local versions and just uploads the final new version of B in the end.

This bundling is realized in the shallot.Operations.Operation class. You get an instance of it in those different ways:

- In the most situations, where filesystem operations potentially make sense, there is an instance directly available as a parameter in a method of your subclasses. The documentation of the Shallot interface classes should help.

- Sometimes there is an instance available, but it is not directly available from the function parameters. In such cases, the documentation should help as well. A typical example is the execution of Actions. Your implementation of shallot.Actions.ActionAction.action gets called with the parameter `info`, which is a shallot.Actions.ExecutionInfo instance. It has the member shallot.Actions.ExecutionInfo.operation which returns a operation object.

- In rare cases you have to explicitely create one. This is when you don't get an existing instance from somewhere, but you need one for some reasones. You can create a new instance with shallot.Operations.Operation.create, but then it must be manually committed with shallot.Operations.Operation.commit in order to transfer all the new versions to their real destinations.

The key method for using the bundling mechanism is shallot.Operations.Operation.fetch_file (and shallot.Operations.Operation.fetch_container_file, which is just a convenience variant with a subtle difference in the arguments). Whenever you need to read the content of a file in order to eventually replace it with a modified version, you should use it.

Please note that operations are also by the only convenient method you would even have whenever your code potentially works with cascades of filesystems. Whenever you need to get the actual data streams of the affected archives for some location like `zip:/[zip:/[ftp://foo/a.zip]//another.zip]//somephoto.jpeg`, calling shallot.Operations.Operation.fetch_file for some part of this address is an easy way (e.g. a shallot.Filesystem.Handler implementation for a custom archive format will need to do that all the time).

For sample code, read the sources of the 'operation' sample plugin (in the Plugin Management).

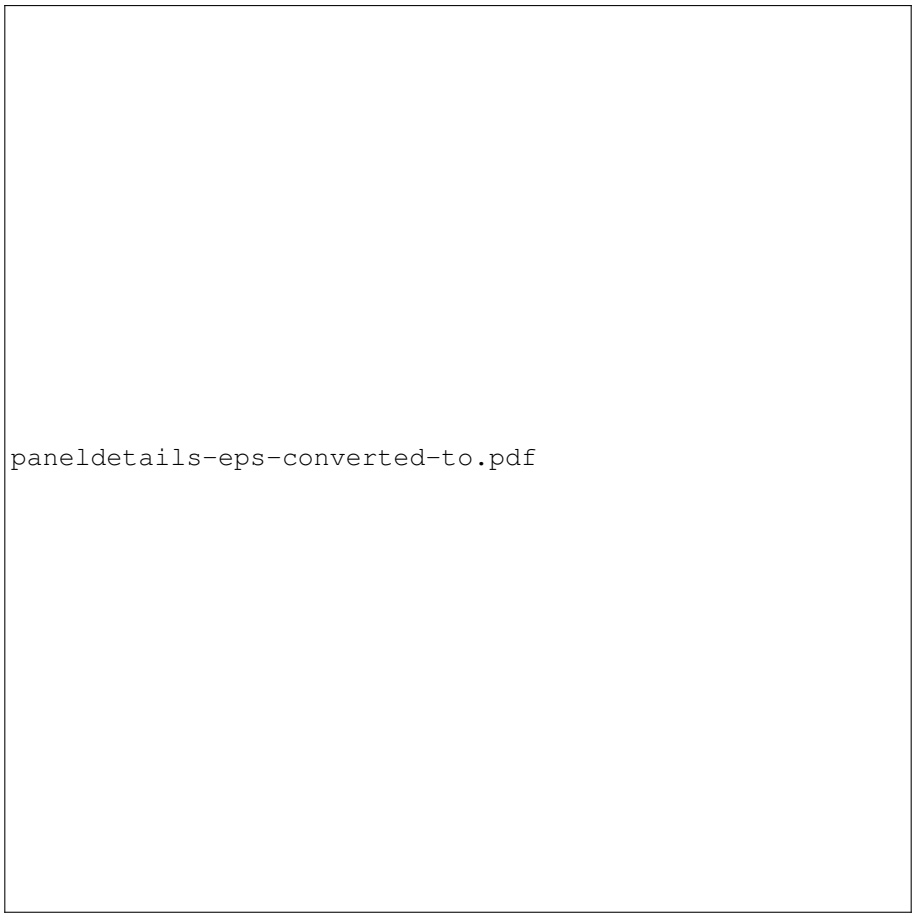
The shallot.Operations.Operation class provide some more. Amongst others, there is the shallot.Operations.Operation.filesystem member, which provides you access to an instance of shallot.Operations.FilesystemOperation. It provides some primitive steps you can also execute in a bundled way. See the class documentation for details.

For sample code, read the sources of the 'filesystemoperations' and 'filesystemoperations.advanced' sample plugins.

The shallot.Operations namespace contains more interesting stuff.

1.1.2.10 Panel Details

Panel details can be provided by a plugin in order to provide the user with additional information about a selection of one or more files in the *Details* part of the Shallot window.



This often is used for presenting some special kinds of file metadata.

The first step to a custom panel detail is to subclass either shallot.PanelDetails.SingleSelectionPanelDetail or shallot.PanelDetails.MultiSelectionPanelDetail. An instance of it must be registered with shallot.PanelDetails.↩ PanelDetail.register.

For sample code, read the sources of the 'paneldetails' sample plugin (in the Plugin Management).

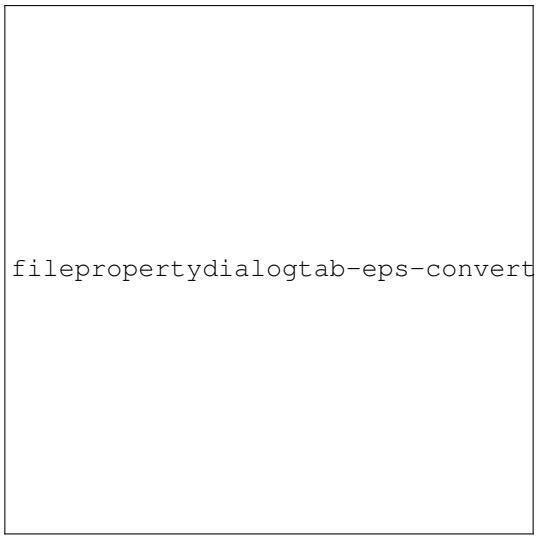
The shallot.PanelDetails namespace contains more interesting stuff.

1.1.2.11 File Property Dialog

The file property dialog is another place where plugins can provide custom pieces of information (and user interactions).

1.1.2.11.1 Dialog Tabs

A custom tab in the property dialog can show one or more information pieces in different views. It can also offer buttons for user interaction.



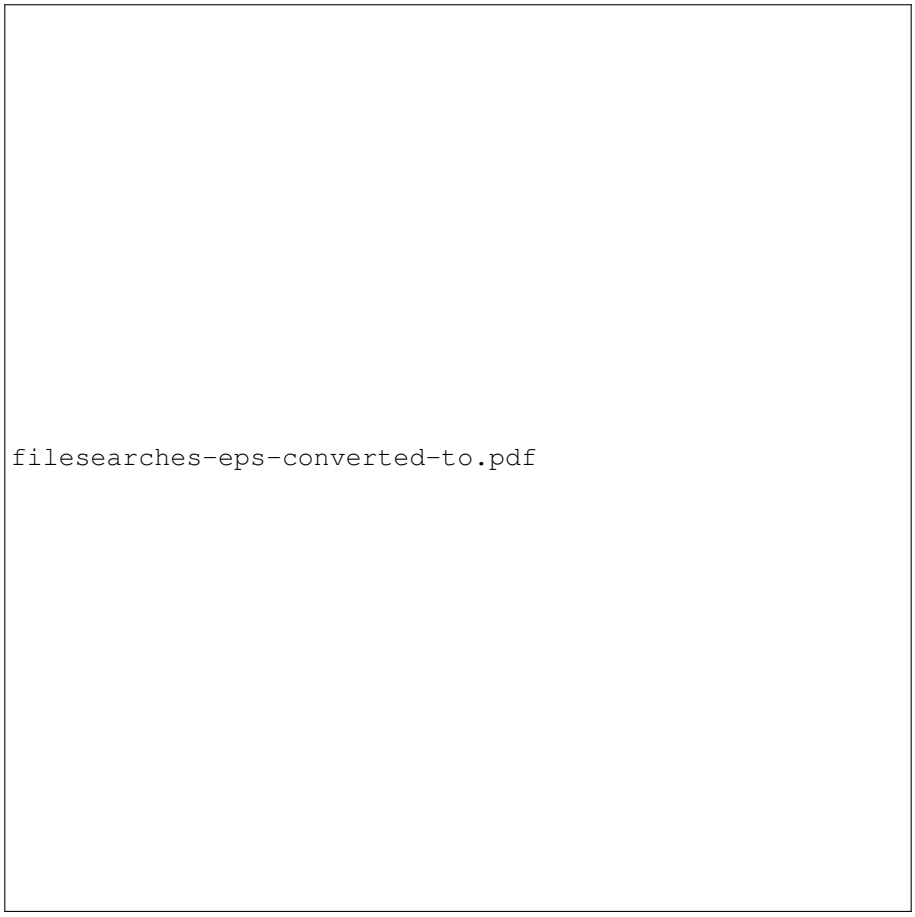
The first step is to subclass shallot.FilePropertyDialog.Tab and to register this class with shallot.FileProperty↩ Dialog.Tab.register.

For sample code, read the sources of the 'filepropertydialogtab' sample plugin (in the Plugin Management).

The shallot.FilePropertyDialog namespace contains more interesting stuff.

1.1.2.12 File Searches

Custom search criterion implementations bring additional functionality to Shallot file searches.



The first step is to subclass shallot.FileSearch.SearchCriterion and also shallot.FileSearch.SearchCriterionFactory (the latter one has to refer to the former one in the constructor call; see class documentations). Register the latter one with shallot.FileSearch.SearchCriterionFactory.register.

For sample code, read the sources of the 'searchcriterion' sample plugin (in the Plugin Management).

The shallot.FileSearch namespace contains more interesting stuff.

1.1.2.13 Thumbnails

Thumbnails are visible in some view modes only. In order to support a new file format or to show certain custom thumbnail content in particular situations, a plugin can add a thumbnail provider implementation to the thumbnail creation mechanism.

The first step is to subclass shallot.ThumbnailProvider and to register an instance of this class with shallot.↩ ThumbnailProvider.register.

For sample code, read the sources of the 'thumbnailprovider' sample plugin (in the Plugin Management).

1.1.2.14 Bookmarks

Bookmarks are shortcuts somewhere in the user interface, which allow the user to fastly jump to some common filesystem locations.

The bookmark menu becomes visible once some bookmarks exist. A plugin can deal with bookmarks as well. It can create and manage them with the methods in shallot.Bookmarks.

For sample code, read the sources of the 'bookmarks' sample plugin (in the Plugin Management).

1.1.2.15 Logging

Plugins can write all kinds of information to the Shallot log in order to find potential issues. See shallot.Logging for details.

1.1.2.16 Utilities

There are some utilities for working with the environment available in shallot.Environment. This includes shallot.↩ Environment.Thread for running code asynchronously, shallot.Environment.Timer for running a code in regular intervals and more.

1.2 Packages

Here are the packages with brief descriptions (if available):

shallot
The Shallot API for Python modules . . . . . ??

1.3 Class Hierarchy

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

shallot._ApiProxy . . . ??
- shallot.Actions.ActionAction . . . ??
- shallot.Actions.ByRegExpPredicate . . . ??
- shallot.Actions.DontResolveLinksPredicate . . . ??
- shallot.Actions.HideOnCurrentDirectoryLevelPredicate . . . ??
- shallot.Actions.HideOnSelectionLevelPredicate . . . ??
- shallot.Actions.KeyShortcutPredicate . . . ??
- shallot.Actions.OnDirectoriesPredicate . . . ??
- shallot.Actions.OnFilesPredicate . . . ??
- shallot.Actions.OnLinksPredicate . . . ??
- shallot.Actions.OnSingleEntrySelectionPredicate . . . ??
- shallot.Actions.PositionIndexPredicate . . . ??
- shallot.Actions.SubmenuAction . . . ??
- shallot.ConfigurationValue . . . ??
- shallot.DetailColumn . . . ??
- shallot.Environment.Thread . . . ??
- shallot.Environment.Timer . . . ??
- shallot.FilePropertyDialog.Tab . . . ??
- shallot.FileSearch.SearchCriterion . . . ??
- shallot.FileSearch.SearchCriterionFactory . . . ??
- shallot.Filesystem.Handler . . . ??
- shallot.Operations.FilesystemOperationProgressMonitor . . . ??
- shallot.PanelDetails.MultiSelectionPanelDetail . . . ??
- shallot.PanelDetails.SingleSelectionPanelDetail . . . ??
- shallot.Setting . . . ??
- shallot.Streaming.ReadDataDevice . . . ??
  - shallot.Streaming.BlobReadDataDevice . . . ??
  - shallot.Streaming.StreamReadDataDevice . . . ??
  - shallot.Streaming.ThreadedReadDataDevice . . . ??
- shallot.ThumbnailProvider . . . ??

shallot.Actions.AbstractAction . . . ??
- shallot.Actions.ActionAction . . . ??
- shallot.Actions.SubmenuAction . . . ??

shallot.PanelDetails.AbstractPanelDetailValueElement . . . ??
- shallot.PanelDetails.PanelDetailValueElementButton . . . ??
- shallot.PanelDetails.PanelDetailValueElementIcon . . . ??
- shallot.PanelDetails.PanelDetailValueElementString . . . ??
- shallot.PanelDetails.PanelDetailValueElementWaiting . . . ??

shallot.Actions . . . ??
shallot.Bookmarks.Bookmark . . . ??
shallot.Bookmarks . . . ??
shallot.ConfigurationValue.Category . . . ??
shallot.Operations.FilesystemOperationStep.ConflictResolution . . . ??
shallot.Actions.DefaultPrecedenceValues . . . ??
shallot.Environment . . . ??
shallot.Eurl . . . ??

Exception
- shallot.Exceptions.ScriptedException . . . ??
  - shallot.Exceptions.ProgramException . . . ??
    - shallot.Exceptions.ArgumentException . . . ??
  - shallot.Exceptions.RuntimeException . . . ??
    - shallot.Exceptions.IOException . . . ??

shallot.Exceptions . . . ??
shallot.Actions.ExecutionInfo . . . ??
shallot.Actions.ExecutionUserFeedback . . . ??
shallot.FilePropertyDialog . . . ??
shallot.FileSearch . . . ??
shallot.Filesystem . . . ??
shallot.Operations.FilesystemOperation . . . ??
shallot.Operations.FilesystemOperationStep . . . ??
shallot.Setting.GroupInfo . . . ??
shallot.Operations.HandlerTransfer . . . ??
shallot.IntlStringMap.IntlString . . . ??
shallot.IntlStringMap . . . ??
shallot.Logging . . . ??
shallot.MainWindow . . . ??
shallot.Actions.ExecutionUserFeedback.MessageBoxButton . . . ??
shallot.Filesystem.Node . . . ??
shallot.Filesystem.NodeList . . . ??
shallot.Filesystem.NodeType . . . ??
shallot.Operations.Operation . . . ??
shallot.Operations . . . ??
shallot.PanelDetails.PanelDetail . . . ??
shallot.PanelDetails.MultiSelectionPanelDetail . . . ??
shallot.PanelDetails.SingleSelectionPanelDetail . . . ??
shallot.PanelDetails.PanelDetailInst . . . ??
shallot.PanelDetails . . . ??
shallot.Actions.Predicate . . . ??
shallot.Actions.ByRegExpPredicate . . . ??
shallot.Actions.DontResolveLinksPredicate . . . ??
shallot.Actions.HideOnCurrentDirectoryLevelPredicate . . . ??
shallot.Actions.HideOnSelectionLevelPredicate . . . ??
shallot.Actions.KeyShortcutPredicate . . . ??
shallot.Actions.OnDirectoriesPredicate . . . ??
shallot.Actions.OnFilesPredicate . . . ??
shallot.Actions.OnLinksPredicate . . . ??
shallot.Actions.OnSingleEntrySelectionPredicate . . . ??
shallot.Actions.PositionIndexPredicate . . . ??
shallot.FilePropertyDialog.Tab.PropertyButtonConfig . . . ??
shallot.FilePropertyDialog.Tab.PropertyConfig . . . ??
shallot.FilePropertyDialog.Tab.PropertyType . . . ??
shallot.Streaming . . . ??
shallot.FilePropertyDialog.TabPropertyIconTextBanner . . . ??

1.4 Class List

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

shallot._ApiProxy
Common superclass for many classes in this interface . . . ??
shallot.Actions.AbstractAction
Abstract base class for executable actions or submenu structures of them . . . ??
shallot.PanelDetails.AbstractPanelDetailValueElement
Abstract base class for an element of a panel detail row's value . . . ??
shallot.Actions.ActionAction
Abstract base class for an executable action, which can be made visible in menus or the toolbar or executed directly . . . ??

shallot.Actions
Everything about actions . . . ??
shallot.Exceptions.ArgumentException
Shallot exception for failed operation due to invalid arguments given to some program part . . ??
shallot.Streaming.BlobReadDataDevice
A binary source backed by static binary content in a byte array . . . ??
shallot.Bookmarks.Bookmark
A bookmark . . . ??
shallot.Bookmarks
Everything about Bookmarks . . . ??
shallot.Actions.ByRegExpPredicate
Shows actions only when the selection paths matches a regular expression . . . ??
shallot.ConfigurationValue.Category
Categories of shallot.ConfigurationValue implementations . . . ??
shallot.ConfigurationValue
Abstract base class for a configuration value . . . ??
shallot.Operations.FilesystemOperationStep.ConflictResolution
Enumeration of conflict resolution strategies . . . ??
shallot.Actions.DefaultPrecedenceValues
Reference values for calculating default precedence values of open actions . . . ??
shallot.DetailColumn
Abstract base class for a detail column for shallot.Filesystem.Node instances . . . ??
shallot.Actions.DontResolveLinksPredicate
Disables links resolving . . . ??
shallot.Environment
Environment information and some utilities . . . ??
shallot.Eurl
A EURL . . . ??
shallot.Exceptions
Everything about exceptions . . . ??
shallot.Actions.ExecutionInfo
An object for signalling action execution state changes between an action implementation and the Shallot core (mostly to the core) . . . ??
shallot.Actions.ExecutionUserFeedback
An object for communication with the user in action implementations . . . ??
shallot.FilePropertyDialog
The file property dialog . . . ??
shallot.FileSearch
File searches . . . ??
shallot.Filesystem
The filesystem . . . ??
shallot.Operations.FilesystemOperation
A high-level provider of filesystem operations . . . ??
shallot.Operations.FilesystemOperationProgressMonitor
Abstract base class for progress monitors, used for monitoring progress of some operations in shallot.Operations.FilesystemOperation and for conflict resolution . . . ??
shallot.Operations.FilesystemOperationStep
One step in a larger filesystem transfer running inside some operations in shallot.Operations.↩ FilesystemOperation . . . ??
shallot.Setting.GroupInfo
Enumeration of groups to which a setting can belong . . . ??
shallot.Filesystem.Handler
Abstract base class for a custom filesystem handler . . . ??
shallot.Operations.HandlerTransfer
Used for some additional functionality in some transfer operations in shallot.Filesystem.Handler instances . . . ??
shallot.Actions.HideOnCurrentDirectoryLevelPredicate
Shows actions only when not searching for 'current directory level' actions . . . ??

shallot.Actions.HideOnSelectionLevelPredicate
Shows actions only when not searching for 'selection level' actions . . . ??
shallot.IntlStringMap.IntlString
A multi-language string . . . ??
shallot.IntlStringMap
A multi-language string map used for localization of plugins . . . ??
shallot.Exceptions.IOException
Shallot exception in IO . . . ??
shallot.Actions.KeyShortcutPredicate
Sets a keyboard shortcut . . . ??
shallot.Logging
Methods for logging . . . ??
shallot.MainWindow
A Shallot main window . . . ??
shallot.Actions.ExecutionUserFeedback.MessageBoxButton
Buttons in a message box from shallot.Actions.ExecutionUserFeedback . . . ??
shallot.PanelDetails.MultiSelectionPanelDetail
Abstract base class for a detail panel entry, which occurs when multiple items are selected . . . ??
shallot.Filesystem.Node
A model representation of a location in the filesystem tree . . . ??
shallot.Filesystem.NodeList
A list editor for a filesystem node list . . . ??
shallot.Filesystem.NodeType
Enumeration of types a shallot.Filesystem.Node can have . . . ??
shallot.Actions.OnDirectoriesPredicate
Shows actions only on directories . . . ??
shallot.Actions.OnFilesPredicate
Shows actions only on files . . . ??
shallot.Actions.OnLinksPredicate
Shows actions only on links . . . ??
shallot.Actions.OnSingleEntrySelectionPredicate
Shows actions only on single-entry selections . . . ??
shallot.Operations.Operation
Transactional read and write filesystem accesses . . . ??
shallot.Operations
Everything about operations on the filesystem . . . ??
shallot.PanelDetails.PanelDetail
Abstract base class for a detail panel entry . . . ??
shallot.PanelDetails.PanelDetailInst
The storage for detail value rows, which are the actual content of a shallot.PanelDetails.Panel↩
Detail for an actual file selection . . . ??
shallot.PanelDetails
Everything about panel details . . . ??
shallot.PanelDetails.PanelDetailValueElementButton
A link button (for executing some code on user behalf) as element of a panel detail row's value ??
shallot.PanelDetails.PanelDetailValueElementIcon
An icon as element of a panel detail row's value . . . ??
shallot.PanelDetails.PanelDetailValueElementString
A piece of text as element of a panel detail row's value . . . ??
shallot.PanelDetails.PanelDetailValueElementWaiting
A 'waiting' placeholder as element of a panel detail row's value . . . ??
shallot.Actions.PositionIndexPredicate
Sets a positioning information index . . . ??
shallot.Actions.Predicate
Controls when and how an action is created . . . ??
shallot.Exceptions.ProgramException
Shallot exception for program errors (typically logical stuff) in the program or plugin . . . ??

shallot.FilePropertyDialog.Tab.PropertyButtonConfig
Definitions for buttons in a shallot.FilePropertyDialog.Tab.PropertyConfig . . . . . ??
shallot.FilePropertyDialog.Tab.PropertyConfig
Definitions for properties, which present a piece of information in a shallot.FilePropertyDialog.Tab ??
shallot.FilePropertyDialog.Tab.PropertyType
A type of a single property . . . . . ??
shallot.Streaming.ReadDataDevice
A scripted source of binary content . . . . . ??
shallot.Exceptions.RuntimeException
Shallot exception for failed operation due to (often external) runtime effects . . . . . ??
shallot.Exceptions.ScriptedException
The Shallot exception class . . . . . ??
shallot.FileSearch.SearchCriterion
Abstract base class for a search criterion . . . . . ??
shallot.FileSearch.SearchCriterionFactory
A factory for a shallot.FileSearch.SearchCriterion class . . . . . ??
shallot.Setting
Abstract base class for a scripted Shallot setting (those who have to be stored manually) . . . ??
shallot.PanelDetails.SingleSelectionPanelDetail
Abstract base class for a detail panel entry, which occurs when one item is selected . . . . . ??
shallot.Streaming
Streams of binary data . . . . . ??
shallot.Streaming.StreamReadDataDevice
A binary source backed by a file object . . . . . ??
shallot.Actions.SubmenuAction
Abstract base class for a submenu action, which can be made visible in menus or the toolbar . ??
shallot.FilePropertyDialog.Tab
Abstract base class for a tab in the Properties dialog . . . . . ??
shallot.FilePropertyDialog.TabPropertyIconTextBanner
Represents values for image/text banners as used for shallot.FilePropertyDialog.Tab.Property↩
Type.IconTextBanner . . . . . ??
shallot.Environment.Thread
A thread executes code asynchronously . . . . . ??
shallot.Streaming.ThreadedReadDataDevice
A binary source dynamically created piecewise in a separate thread . . . . . ??
shallot.ThumbnailProvider
Abstract base class for a thumbnail provider . . . . . ??
shallot.Environment.Timer
A timers executes some code recurringly in some time interval . . . . . ??

1.5 shallot Namespace Reference

The Shallot API for Python modules.

Classes

- class _ApiProxy

  Common superclass for many classes in this interface.
- class Actions

  Everything about actions.
- class Bookmarks

  Everything about Bookmarks.
- class ConfigurationValue

Abstract base class for a configuration value.

- class DetailColumn

  Abstract base class for a detail column for shallot.Filesystem.Node instances.
- class Environment

  Environment information and some utilities.
- class Eurl

  A EURL.
- class Exceptions

  Everything about exceptions.
- class FilePropertyDialog

  The file property dialog.
- class FileSearch

  File searches.
- class Filesystem

  The filesystem.
- class IntlStringMap

  A multi-language string map used for localization of plugins.
- class Logging

  Methods for logging.
- class MainWindow

  A Shallot main window.
- class Operations

  Everything about operations on the filesystem.
- class PanelDetails

  Everything about panel details.
- class Setting

  Abstract base class for a scripted Shallot setting (those who have to be stored manually).
- class Streaming

  Streams of binary data.
- class ThumbnailProvider

  Abstract base class for a thumbnail provider.

1.5.1 Detailed Description

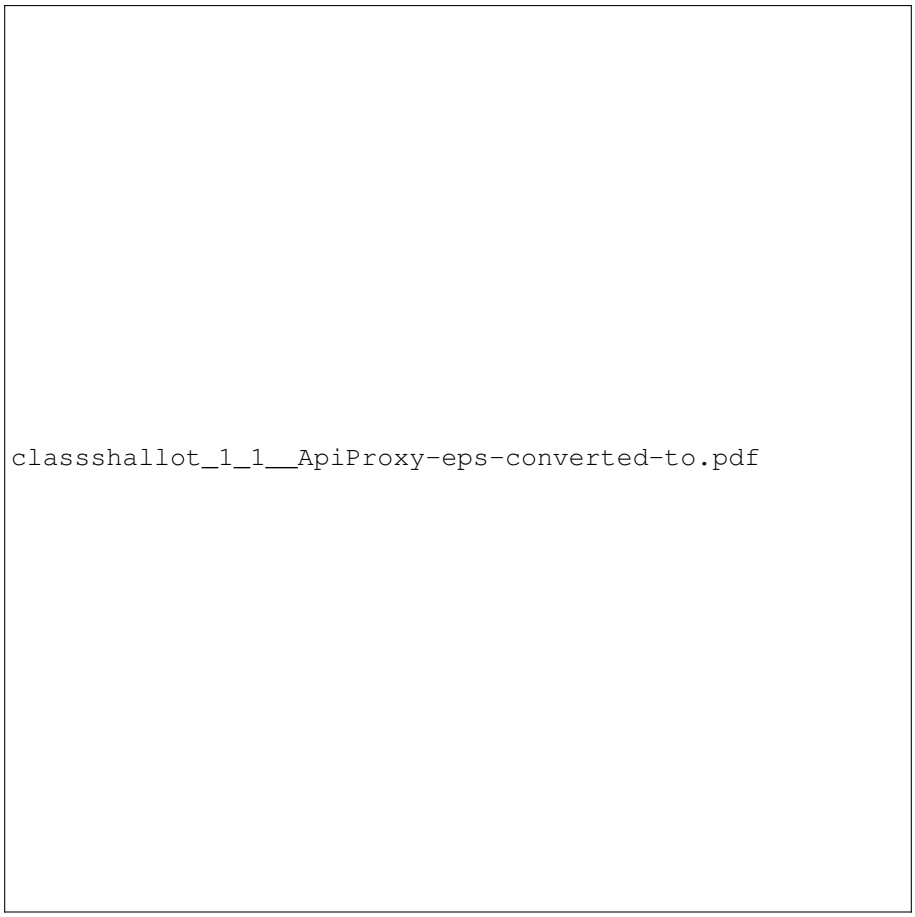
The Shallot API for Python modules.

The manual about the Shallot scripting API gives a full reference.

1.6 shallot._ApiProxy Class Reference

Common superclass for many classes in this interface.

Inheritance diagram for shallot._ApiProxy:



Public Member Functions

- def **__init__** (self, native)

1.6.1 Detailed Description

Common superclass for many classes in this interface.

This class mostly has internal meaning and does not directly offer any services to the plugin developer. Read more.

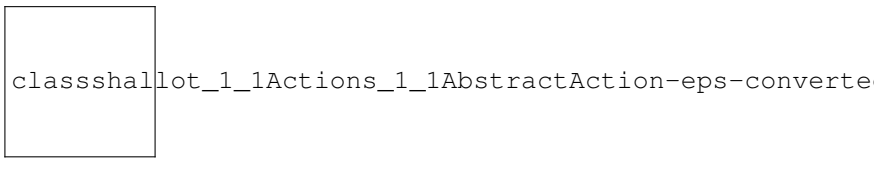
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.7 shallot.Actions.AbstractAction Class Reference

Abstract base class for executable actions or submenu structures of them.

Inheritance diagram for shallot.Actions.AbstractAction:



Public Member Functions

- def initialize (self)

  Initialize the action.
- def set_enabled (self, value)

  Sets if the item is enabled.
- def set_visible (self, value)

  Sets the visibility of this item.
- def enabled (self)

  Checks if this action is enabled.
- def visible (self)

  Checks the visibility of this item (non-recursively).

1.7.1 Detailed Description

Abstract base class for executable actions or submenu structures of them.

See the subclasses of this class. They can be registered with shallot.Actions.register. They can be returned from shallot.Filesystem.Handler.get_actions.

1.7.2 Member Function Documentation

1.7.2.1 initialize()

```
def shallot.Actions.AbstractAction.initialize (
              self )
```

Initialize the action.

This should make the time-consuming parts, e.g. for determining a label or enabled state.

Override this method in custom subclasses or leave the default implementation.

1.7.2.2 set_enabled()

```
def shallot.Actions.AbstractAction.set_enabled (
              self,
              value )
```

Sets if the item is enabled.

Parameters

value	The new value.

1.7.2.3 set_visible()

```
def shallot.Actions.AbstractAction.set_visible (
              self,
              value )
```

Sets the visibility of this item.

Parameters

value	The new value.

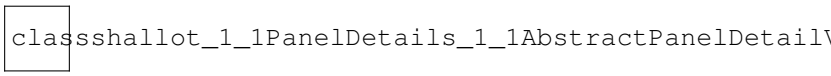
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.8 shallot.PanelDetails.AbstractPanelDetailValueElement Class Reference

Abstract base class for an element of a panel detail row's value.

Inheritance diagram for shallot.PanelDetails.AbstractPanelDetailValueElement:



Public Member Functions

- def **__init__** (self, type, value)

Public Attributes

- **type**
- **value**

1.8.1 Detailed Description

Abstract base class for an element of a panel detail row's value.

This can be something like a text, an icon or a link button. See the subclasses.

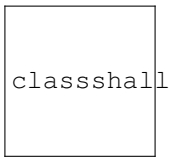
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.9 shallot.Actions.ActionAction Class Reference

Abstract base class for an executable action, which can be made visible in menus or the toolbar or executed directly.

Inheritance diagram for shallot.Actions.ActionAction:



Public Member Functions

- def __init__ (self, text, enabled=True, icon="", defaultActionPrecedence=0)
- def action (self, info)

  The action implementation, i.e.
- def execute (self)

  Executes this action.
- def initialize_sync (self)

  Initializes the action.
- def initialize (self)

  Initialize the action.
- def set_enabled (self, value)

  Sets if the item is enabled.
- def set_visible (self, value)

  Sets the visibility of this item.
- def enabled (self)

  Checks if this action is enabled.
- def visible (self)

  Checks the visibility of this item (non-recursively).

1.9.1 Detailed Description

Abstract base class for an executable action, which can be made visible in menus or the toolbar or executed directly.

See shallot.Actions.AbstractAction for more.

1.9.2 Constructor & Destructor Documentation

1.9.2.1 __init__()

```
def shallot.Actions.ActionAction.__init__ (
              self,
              text,
              enabled = True,
              icon = "",
              defaultActionPrecedence = 0 )
```

Parameters

text	A label text (can be "" if you want to directly execute it instead of adding it to some menu structure).
enabled	If this action is enabled.
icon	Name of an icon.
defaultActionPrecedence	Precedence value for being a default action. This int value must be higher than all others for becoming the default action. See shallot.Actions.DefaultPrecedenceValues.

1.9.3 Member Function Documentation

1.9.3.1 action()

```
def shallot.Actions.ActionAction.action (
              self,
              info )
```

The action implementation, i.e.

what the actions should actually do.

Override this method in custom subclasses.

Parameters

info	A shallot.Actions.ExecutionInfo execution info object.

1.9.3.2 initialize()

```
def shallot.Actions.AbstractAction.initialize (
              self )  [inherited]
```

Initialize the action.

This should make the time-consuming parts, e.g. for determining a label or enabled state.

Override this method in custom subclasses or leave the default implementation.

1.9.3.3 initialize_sync()

```
def shallot.Actions.ActionAction.initialize_sync (
              self )
```

Initializes the action.

This can be called from outside in order to do the initialization.

1.9.3.4 set_enabled()

```
def shallot.Actions.AbstractAction.set_enabled (
              self,
              value )  [inherited]
```

Sets if the item is enabled.

Parameters

value	The new value.

1.9.3.5 set_visible()

```
def shallot.Actions.AbstractAction.set_visible (
              self,
              value )  [inherited]
```

Sets the visibility of this item.

Parameters

value	The new value.

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

- /home/pino/projects/shallot/shallot.py

1.10 shallot.Actions Class Reference

Everything about actions.

Classes

- class AbstractAction

Abstract base class for executable actions or submenu structures of them.

- class ActionAction

  Abstract base class for an executable action, which can be made visible in menus or the toolbar or executed directly.
- class ByRegExpPredicate

  Shows actions only when the selection paths matches a regular expression.
- class DefaultPrecedenceValues

  Reference values for calculating default precedence values of open actions.
- class DontResolveLinksPredicate

  Disables links resolving.
- class ExecutionInfo

  An object for signalling action execution state changes between an action implementation and the Shallot core (mostly to the core).
- class ExecutionUserFeedback

  An object for communication with the user in action implementations.
- class HideOnCurrentDirectoryLevelPredicate

  Shows actions only when not searching for 'current directory level' actions.
- class HideOnSelectionLevelPredicate

  Shows actions only when not searching for 'selection level' actions.
- class KeyShortcutPredicate

  Sets a keyboard shortcut.
- class OnDirectoriesPredicate

  Shows actions only on directories.
- class OnFilesPredicate

  Shows actions only on files.
- class OnLinksPredicate

  Shows actions only on links.
- class OnSingleEntrySelectionPredicate

  Shows actions only on single-entry selections.
- class PositionIndexPredicate

  Sets a positioning information index.
- class Predicate

  Controls when and how an action is created.
- class SubmenuAction

  Abstract base class for a submenu action, which can be made visible in menus or the toolbar.

Static Public Member Functions

- def register (actiontype, category="manage", predicates=[ ])

  Registers a subclass of shallot.Actions.AbstractAction.

Public Attributes

- **ActionType**

1.10.1 Detailed Description

Everything about actions.

1.10.2 Member Function Documentation

1.10.2.1 register()

```
def shallot.Actions.register (
              actiontype,
              category = "manage",
              predicates = [] )  [static]
```

Registers a subclass of shallot.Actions.AbstractAction.

This makes it permanently visible in the context menu of selections and/or in the toolbar. If and where the action is visible depends on the action implementation (see constructor parameters), the situation (what is selected in the user interface?) and the registration `mode`.

Parameters

actiontype	The class implementing shallot.Actions.AbstractAction. Its constructor must be callable with just `actiontype(nodes)`. The `nodes` parameter contains a list of shallot.Filesystem.Node containing the node selection.
category	The action category. Used for grouping in menus. Typically available are `"create"`, `"open"` and `"manage"`.
predicates	A list of shallot.Actions.Predicate controlling when and how the action is offered.

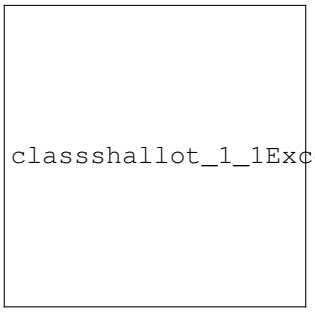
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.11 shallot.Exceptions.ArgumentException Class Reference

Shallot exception for failed operation due to invalid arguments given to some program part.

Inheritance diagram for shallot.Exceptions.ArgumentException:



Public Member Functions

- def __init__ (self, details=None, message=None, isResumeable=True, detailsAreInteresting=None, _↩ class="")

  See shallot.Exceptions.ScriptedException.__init__ for details.
- def isExceptionClass (self, classname)

  Checks if this exception is instance of a given exception class.

1.11.1 Detailed Description

Shallot exception for failed operation due to invalid arguments given to some program part.

It typically allows resume but not retry (special cases may override each of them). Read more about Shallot Exceptions.

1.11.2 Member Function Documentation

1.11.2.1 isExceptionClass()

```
def shallot.Exceptions.ScriptedException.isExceptionClass (
              self,
              classname )  [inherited]
```

Checks if this exception is instance of a given exception class.

Parameters

classname	A exception class name (as string).

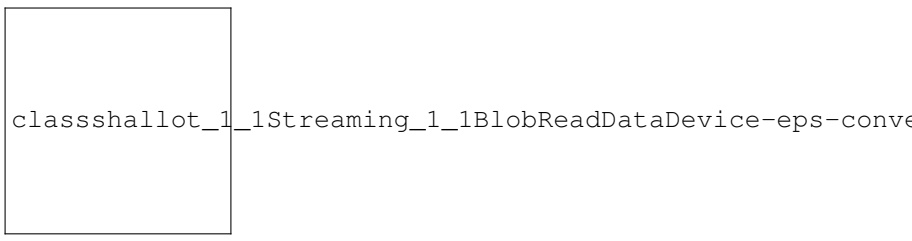
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.12 shallot.Streaming.BlobReadDataDevice Class Reference

A binary source backed by static binary content in a byte array.

Inheritance diagram for shallot.Streaming.BlobReadDataDevice:



Public Member Functions

- def __init__ (self, content)
- def getdata (self)

  See shallot.Streaming.ReadDataDevice.
- def read (self)

  Reads a chuck of data as Python byte array.
- def readall (self)

  Reads the complete data as Python byte array.

Public Attributes

- **content**

1.12.1 Detailed Description

A binary source backed by static binary content in a byte array.

For really large content (many megabytes), you should avoid creating complete copies but use one of the other subclasses or implementing an own one.

1.12.2 Constructor & Destructor Documentation

1.12.2.1 __init__()

```
def shallot.Streaming.BlobReadDataDevice.__init__ (
              self,
              content )
```

Parameters

content	The byte array containing the content.

1.12.3 Member Function Documentation

1.12.3.1 read()

```
def shallot.Streaming.ReadDataDevice.read (
              self )  [inherited]
```

Reads a chuck of data as Python byte array.

An empty array signals the stream ended.

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

- /home/pino/projects/shallot/shallot.py

1.13 shallot.Bookmarks.Bookmark Class Reference

A bookmark.

Public Member Functions

- def __init__ (self)
  Can't be constructed directly.
- def eurl (self)
  The location this bookmark points to.
- def folder (self)
  The subcollection (as list of strings).
- def id (self)
  The bookmark id.
- def label (self)
  A textual description of this bookmark.
- def tags (self)
  The tags strings.

1.13.1 Detailed Description

A bookmark.

Change it with the methods in shallot.Bookmarks.

1.13.2 Member Function Documentation

1.13.2.1 id()

```
def shallot.Bookmarks.Bookmark.id (
              self )
```

The bookmark id.

It is not display anywhere but only used for the management methods in shallot.Bookmarks.

1.13.2.2 tags()

```
def shallot.Bookmarks.Bookmark.tags (
              self )
```

The tags strings.

This value is solely used by plugins for their bookkeeping. It can allow to programmatically find a particular bookmark (if the creator wrote meaningful information in it).

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

- /home/pino/projects/shallot/shallot.py

1.14 shallot.Bookmarks Class Reference

Everything about Bookmarks.

Classes

- class Bookmark

  A bookmark.

Static Public Member Functions

- def get_bookmarks ()

  Returns a list of all existing shallot.Bookmarks.Bookmark instances.
- def has_bookmarks ()

  Returns True if there are any bookmarks stored.
- def add_bookmark (eurl, folder=[ ], label=None, tags=None)

  Adds a new bookmark.
- def remove_bookmark (id)

  Removes a bookmark.
- def change_bookmark (id, eurl, label=None)

  Changes bookmark data.
- def change_bookmark_tags (id, tags)

  Changes bookmark tags.
- def move_bookmark_up (id)

  Moves a bookmark upwards in its collection.
- def move_bookmark_down (id)

  Moves a bookmark downwards in its collection.
- def move_bookmark_to_folder (id, folder)

  Moves a bookmark to another subcollection.

1.14.1 Detailed Description

Everything about Bookmarks.

1.14.2 Member Function Documentation

1.14.2.1 add_bookmark()

```
def shallot.Bookmarks.add_bookmark (
              eurl,
              folder = [],
              label = None,
              tags = None )  [static]
```

Adds a new bookmark.

Parameters

eurl	The location to bookmark.
folder	A string list describing in which subcollection this bookmark should be placed.
label	The textual description of the new bookmark.
tags	A string for internal bookkeeping. Use it for recognizing your own bookmarks later on. See shallot.Bookmarks.Bookmark.tags.

1.14.2.2 change_bookmark()

```
def shallot.Bookmarks.change_bookmark (
              id,
              eurl,
              label = None )  [static]
```

Changes bookmark data.

Parameters

id	The bookmark id this call is about (See shallot.Bookmarks.Bookmark.id).
eurl	The new location.
label	The new textual description.

1.14.2.3 change_bookmark_tags()

```
def shallot.Bookmarks.change_bookmark_tags (
              id,
              tags )  [static]
```

Changes bookmark tags.

See shallot.Bookmarks.Bookmark.tags.

Parameters

id	The bookmark id this call is about (See shallot.Bookmarks.Bookmark.id).
tags	The new tags string.s

1.14.2.4 move_bookmark_down()

```
def shallot.Bookmarks.move_bookmark_down (
              id )  [static]
```

Moves a bookmark downwards in its collection.

Parameters

id	The bookmark id this call is about (See shallot.Bookmarks.Bookmark.id).

1.14.2.5 move_bookmark_to_folder()

```
def shallot.Bookmarks.move_bookmark_to_folder (
              id,
              folder )  [static]
```

Moves a bookmark to another subcollection.

Parameters

id	The bookmark id this call is about (See shallot.Bookmarks.Bookmark.id).
folder	The new subcollection (as list of strings).

1.14.2.6 move_bookmark_up()

```
def shallot.Bookmarks.move_bookmark_up (
              id )  [static]
```

Moves a bookmark upwards in its collection.

Parameters

id	The bookmark id this call is about (See shallot.Bookmarks.Bookmark.id).

1.14.2.7 remove_bookmark()

```
def shallot.Bookmarks.remove_bookmark (
              id )  [static]
```

Removes a bookmark.

Parameters

id	The bookmark id this call is about (See shallot.Bookmarks.Bookmark.id).

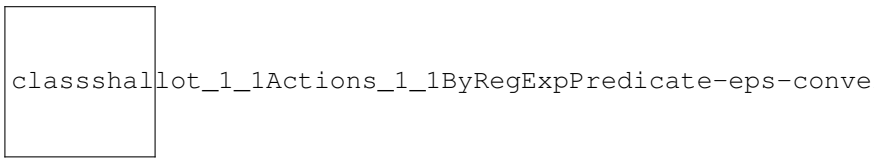
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.15 shallot.Actions.ByRegExpPredicate Class Reference

Shows actions only when the selection paths matches a regular expression.

Inheritance diagram for shallot.Actions.ByRegExpPredicate:



Public Member Functions

- def **__init__** (self, pattern)

1.15.1 Detailed Description

Shows actions only when the selection paths matches a regular expression.

See base class for more information.

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

- /home/pino/projects/shallot/shallot.py

1.16 shallot.ConfigurationValue.Category Class Reference

Categories of shallot.ConfigurationValue implementations.

Static Public Attributes

- GUI = internalstuff.configurationcategory_gui()

  User interface category.
- Behavior = internalstuff.configurationcategory_behavior()

  Behavioral configuration category.
- ExternalTools = internalstuff.configurationcategory_externaltools()

  External tools category.

1.16.1 Detailed Description

Categories of shallot.ConfigurationValue implementations.

They are used for grouping them in the dialogs. There is no difference in behavior implied by this choice.

1.16.2 Member Data Documentation

1.16.2.1 Behavior

```
shallot.ConfigurationValue.Category.Behavior = internalstuff.configurationcategory_behavior()
[static]
```

Behavioral configuration category.

1.16.2.2 ExternalTools

```
shallot.ConfigurationValue.Category.ExternalTools = internalstuff.configurationcategory_↩
externaltools()  [static]
```

External tools category.

1.16.2.3 GUI

```
shallot.ConfigurationValue.Category.GUI = internalstuff.configurationcategory_gui()  [static]
```

User interface category.

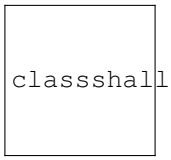
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.17 shallot.ConfigurationValue Class Reference

Abstract base class for a configuration value.

Inheritance diagram for shallot.ConfigurationValue:



Classes

- class Category

  Categories of shallot.ConfigurationValue implementations.

Public Member Functions

- def __init__ (self, name, defaultvalue, category=0, description="", longdescription="", changehint=None)
- def value (self)

  Returns the current value.
- def set_value (self, value)

  Sets the value.

Public Attributes

- **vtype**

1.17.1 Detailed Description

Abstract base class for a configuration value.

1.17.2 Constructor & Destructor Documentation

1.17.2.1 __init__()

```
def shallot.ConfigurationValue.__init__ (
              self,
              name,
              defaultvalue,
              category = 0,
              description = "",
              longdescription = "",
              changehint = None )
```

Parameters

name	The config value name.
defaultvalue	The default value.
category	The category. One of shallot.ConfigurationValue.Category.
description	The short description.
longdescription	The long description.
changehint	A label hint for change buttons in user interfaces.

1.17.3 Member Function Documentation

1.17.3.1 set_value()

```
def shallot.ConfigurationValue.set_value (
              self,
              value )
```

Sets the value.

Parameters

value	The new value.

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

- /home/pino/projects/shallot/shallot.py

1.18 shallot.Operations.FilesystemOperationStep.ConflictResolution Class Reference

Enumeration of conflict resolution strategies.

Static Public Attributes

- MergeDirectories = internalstuff.operationstep_conflictresolution_mergedirectories()
  Merge source into destination directory.
- OverwriteDestination = internalstuff.operationstep_conflictresolution_overwritedestination()
  Overwrite the destination.
- RenameDestinationBefore = internalstuff.operationstep_conflictresolution_renamedestinationbefore()
  Rename the file at the destination before transferring.
- UseDifferentDestinationName = internalstuff.operationstep_conflictresolution_usedifferentdestinationname()
  Transfer to another destination filename.
- Skip = internalstuff.operationstep_conflictresolution_skip()
  Skip this element.
- Unresolved = internalstuff.operationstep_conflictresolution_unresolved()
  No strategy.
- Indirect = internalstuff.operationstep_conflictresolution_indirect()
  Indirectly solved.

1.18.1 Detailed Description

Enumeration of conflict resolution strategies.

1.18.2 Member Data Documentation

1.18.2.1 Indirect

```
shallot.Operations.FilesystemOperationStep.ConflictResolution.Indirect = internalstuff.↩
operationstep_conflictresolution_indirect()  [static]
```

Indirectly solved.

Only set by the engine.

1.18.2.2 MergeDirectories

```
shallot.Operations.FilesystemOperationStep.ConflictResolution.MergeDirectories = internalstuff.↩
operationstep_conflictresolution_mergedirectories()  [static]
```

Merge source into destination directory.

1.18.2.3 OverwriteDestination

```
shallot.Operations.FilesystemOperationStep.ConflictResolution.OverwriteDestination = internalstuff.↩
operationstep_conflictresolution_overwritedestination()  [static]
```

Overwrite the destination.

1.18.2.4 RenameDestinationBefore

```
shallot.Operations.FilesystemOperationStep.ConflictResolution.RenameDestinationBefore = internalstuff.↩
operationstep_conflictresolution_renamedestinationbefore()  [static]
```

Rename the file at the destination before transferring.

1.18.2.5 Skip

```
shallot.Operations.FilesystemOperationStep.ConflictResolution.Skip = internalstuff.operationstep↩
_conflictresolution_skip()  [static]
```

Skip this element.

1.18.2.6 Unresolved

```
shallot.Operations.FilesystemOperationStep.ConflictResolution.Unresolved = internalstuff.↩
operationstep_conflictresolution_unresolved()  [static]
```

No strategy.

1.18.2.7 UseDifferentDestinationName

```
shallot.Operations.FilesystemOperationStep.ConflictResolution.UseDifferentDestinationName =
internalstuff.operationstep_conflictresolution_usedifferentdestinationname()  [static]
```

Transfer to another destination filename.

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

- /home/pino/projects/shallot/shallot.py

1.19 shallot.Actions.DefaultPrecedenceValues Class Reference

Reference values for calculating default precedence values of open actions.

Static Public Attributes

- OpenFile = internalstuff.actiondefaultprecedencevalues_openfile()

  Normal file open action.
- BuiltinSpecialOpen = internalstuff.actiondefaultprecedencevalues_builtinspecialopen()

  Special open action.

1.19.1 Detailed Description

Reference values for calculating default precedence values of open actions.

1.19.2 Member Data Documentation

1.19.2.1 BuiltinSpecialOpen

```
shallot.Actions.DefaultPrecedenceValues.BuiltinSpecialOpen = internalstuff.actiondefaultprecedencevalues↩
_builtinspecialopen()  [static]
```

Special open action.

1.19.2.2 OpenFile

```
shallot.Actions.DefaultPrecedenceValues.OpenFile = internalstuff.actiondefaultprecedencevalues↩
_openfile()  [static]
```

Normal file open action.

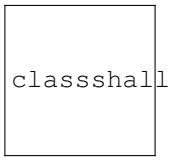
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.20 shallot.DetailColumn Class Reference

Abstract base class for a detail column for shallot.Filesystem.Node instances.

Inheritance diagram for shallot.DetailColumn:



Public Member Functions

- def __init__ (self, name, displayname, positionGroup=None, positionIndex=None, sort_doTypediff=True, defaultWidth=-1, isRightAligned=False)
- def determine_value (self, node, operation)

  Determines the column value for a node.
- def apply_value (self, eurl, operation, value)

  Set the detail value for a given eurl.
- def compute_value (self, node, operation)

  Queries the column value for a node.

Static Public Member Functions

- def find_by_name (name)

  Finds a shallot.DetailColumn implementation by name.
- def register_as_transferrable (index, detailcolumn)

  Registers a shallot.DetailColumn for transferring those details in file transfers (e.g.

Static Public Attributes

- INDEX_CORE = internalstuff.displayindex_core()

  A integer value to be used as base display index for very interesting columns.
- INDEX_INTERESTING = internalstuff.displayindex_interesting()

  A integer value to be used as base display index for interesting columns.
- INDEX_EXOTIC = internalstuff.displayindex_exotic()

  A integer value to be used as base display index for not generally interesting columns.

1.20.1 Detailed Description

Abstract base class for a detail column for shallot.Filesystem.Node instances.

Those can e.g. be seen in the file list view, but can also be used internally by other places.

It encapsulates the retrieval logic and metadata for one piece of additional information a shallot.Filesystem.Node can have (e.g. the filesize). Retrieving the values is designed to be asynchronous. Each instance represents one column, while the actual logic is implemented in subclasses. For a new detail column, subclass this class and implement at least shallot.DetailColumn.determine_value.

1.20.2 Constructor & Destructor Documentation

1.20.2.1 __init__()

```
def shallot.DetailColumn.__init__ (
              self,
              name,
              displayname,
              positionGroup = None,
              positionIndex = None,
              sort_doTypediff = True,
              defaultWidth = -1,
              isRightAligned = False )
```

Parameters

name	The internal name (must be unique).
displayname	The label text.
positionGroup	Controls display order. See Position Indexes for details. Use one of the `INDEX_*` values from shallot.DetailColumn.
positionIndex	Controls display order. See Position Indexes for details.
sort_doTypediff	Shall sorting differ between files and directories?
defaultWidth	The default width in pixels (optional).
isRightAligned	If the column values are right aligned (optional).

1.20.3 Member Function Documentation

1.20.3.1 apply_value()

```
def shallot.DetailColumn.apply_value (
              self,
              eurl,
              operation,
              value )
```

Set the detail value for a given eurl.

Used for transferring details in file transfers (see shallot.DetailColumn.register_as_transferrable).

Override this method in custom subclasses or leave the default implementation.

Parameters

eurl	The shallot.Eurl for which the value must be determined.
operation	The shallot.Operations.Operation operation object.
value	The detail value (as string).

1.20.3.2 compute_value()

```
def shallot.DetailColumn.compute_value (
              self,
              node,
              operation )
```

Queries the column value for a node.

Parameters

node	The shallot.Filesystem.Node node for which the value must be determined.
operation	The shallot.Operations.Operation operation object.

1.20.3.3 determine_value()

```
def shallot.DetailColumn.determine_value (
              self,
```

```
                node,
                operation )
```

Determines the column value for a node.

Override this method in custom subclasses. Only used internally. For querying foreign detail columns, use compute_value instead.

Parameters

node	The shallot.Filesystem.Node node for which the value must be determined.
operation	The shallot.Operations.Operation operation object.

1.20.3.4 find_by_name()

```
def shallot.DetailColumn.find_by_name (
                name )  [static]
```

Finds a shallot.DetailColumn implementation by name.

Parameters

name	The detail column name.

1.20.3.5 register_as_transferrable()

```
def shallot.DetailColumn.register_as_transferrable (
                index,
                detailcolumn )  [static]
```

Registers a shallot.DetailColumn for transferring those details in file transfers (e.g.

when the user copies files). This requires to implement some methods.

Parameters

index	An integer which controls the order of transferring.
detailcolumn	The shallot.DetailColumn detail column.

1.20.4 Member Data Documentation

1.20.4.1 INDEX_CORE

```
shallot.DetailColumn.INDEX_CORE = internalstuff.displayindex_core()  [static]
```

A integer value to be used as base display index for very interesting columns.

1.20.4.2 INDEX_EXOTIC

```
shallot.DetailColumn.INDEX_EXOTIC = internalstuff.displayindex_exotic()  [static]
```

A integer value to be used as base display index for not generally interesting columns.

1.20.4.3 INDEX_INTERESTING

```
shallot.DetailColumn.INDEX_INTERESTING = internalstuff.displayindex_interesting()  [static]
```

A integer value to be used as base display index for interesting columns.

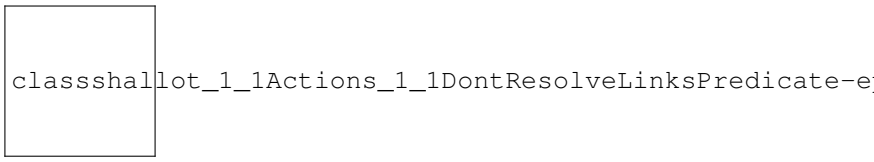
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.21 shallot.Actions.DontResolveLinksPredicate Class Reference

Disables links resolving.

Inheritance diagram for shallot.Actions.DontResolveLinksPredicate:



Public Member Functions

- def **__init__** (self)

1.21.1 Detailed Description

Disables links resolving.

See base class for more information.

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

- /home/pino/projects/shallot/shallot.py

1.22 shallot.Environment Class Reference

Environment information and some utilities.

Classes

- class Thread

  A thread executes code asynchronously.
- class Timer

  A timers executes some code recurringly in some time interval.

Static Public Attributes

- is_debug_build = internalstuff.is_debug_build()

  If Shallot was built in debug mode.
- shallot_data_dir = internalstuff.shallotDataDir()

  The directory containing the shallot data files.
- shallot_builtinplugin_dir = internalstuff.shallotBuiltinPluginDir()

  The directory containing the shallot builtin plugin files.
- shallot_systemplugin_dir = internalstuff.shallotSystemPluginDir()

  The directory containing the shallot system plugin files.
- shallot_userplugin_dir = internalstuff.shallotUserPluginDir()

  The directory containing the shallot user plugin files.
- shallot_language = internalstuff.shallotLanguage()

  The language code of the current user interface language.

1.22.1 Detailed Description

Environment information and some utilities.

1.22.2 Member Data Documentation

1.22.2.1 is_debug_build

```
shallot.Environment.is_debug_build = internalstuff.is_debug_build()  [static]
```

If Shallot was built in debug mode.

1.22.2.2 shallot_builtinplugin_dir

```
shallot.Environment.shallot_builtinplugin_dir = internalstuff.shallotBuiltinPluginDir()  [static]
```

The directory containing the shallot builtin plugin files.

1.22.2.3 shallot_data_dir

```
shallot.Environment.shallot_data_dir = internalstuff.shallotDataDir()  [static]
```

The directory containing the shallot data files.

1.22.2.4 shallot_language

```
shallot.Environment.shallot_language = internalstuff.shallotLanguage()  [static]
```

The language code of the current user interface language.

1.22.2.5 shallot_systemplugin_dir

```
shallot.Environment.shallot_systemplugin_dir = internalstuff.shallotSystemPluginDir()  [static]
```

The directory containing the shallot system plugin files.

1.22.2.6 shallot_userplugin_dir

```
shallot.Environment.shallot_userplugin_dir = internalstuff.shallotUserPluginDir()  [static]
```

The directory containing the shallot user plugin files.

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

- /home/pino/projects/shallot/shallot.py

1.23 shallot.Eurl Class Reference

A EURL.

Public Member Functions

- def __init__ (self)

  Can't be constructed directly.
- def as_string (self)

  Returns the textual value.
- def **__str__** (self)
- def basename (self)

  Returns the last path segment.
- def parent_segment (self)

  Returns the parent shallot.Eurl.
- def with_appended_segment (self, segment)

  Returns a new shallot.Eurl from this one with "/basename" appended.
- def with_appended_segments (self, path)

  Returns a new shallot.Eurl with path segments "/pa/t/h/" appended.
- def enwrap_with_outer_url (self, scheme, hostname, path)

  Returns a new shallot.Eurl containing this one packed as embedding and new outer parts `scheme`, `hostname` and `path`.
- def path (self)

  Returns the path part (from the outer url of this shallot.Eurl).
- def root (self)

  Returns the root shallot.Eurl from this one.
- def hostname (self)

  Returns the hostname part (from the outer url of this shallot.Eurl).
- def scheme (self)

  Returns the scheme (from the outer url of this shallot.Eurl).
- def outer_url (self)

  Returns a new shallot.Eurl containing only the outer part of this one (strips the embeddings).
- def outermost_inner_eurl (self)

  Returns a new shallot.Eurl containing only the embedding of this one.
- def has_inner_urls (self)

  Checks if this shallot.Eurl has embeddings.
- def outer_url_is_root_directory (self)

  Checks if this shallot.Eurl is a root path (with or without embeddings).
- def has_parent_segment (self)

  Checks if this shallot.Eurl has a parent segment.
- def is_prefix_of (self, eurl)

  Checks if this shallot.Eurl is a prefix of another one.

Static Public Member Functions

- def from_string (s)

  Returns a shallot.Eurl from a string.
- def create (scheme, hostname, path)

  Returns a shallot.Eurl from three parts.

1.23.1 Detailed Description

A EURL.

You can get instances with global methods like shallot.Eurl.create and shallot.Eurl.from_string.

Please note: Instances can be associated with any kind of elements in the filesystem (files, directories, links, ...). It might also point to something which does not exist at all. The documentation sometimes explicitly makes a difference between those kinds (files, directories, links, ...; often called 'node type'). But it often uses the term 'file' implicitly while meaning all kinds of elements; assuming that e.g. a directory is just a special kind of a file. It should be clear from the particular context which meaning applies.

1.23.2 Member Function Documentation

1.23.2.1 as_string()

```
def shallot.Eurl.as_string (
              self )
```

Returns the textual value.

This is what shallot.Eurl.from_string would expect as parameter.

1.23.2.2 basename()

```
def shallot.Eurl.basename (
              self )
```

Returns the last path segment.

This is the text behind the last slash. Examples: `"baz"` for `file:///foo/bar/baz`. `""` for `file:///`.

1.23.2.3 create()

```
def shallot.Eurl.create (
              scheme,
              hostname,
              path )  [static]
```

Returns a shallot.Eurl from three parts.

The structure is `scheme://hostname/pa/th`.

Parameters

scheme	The scheme part.
hostname	The hostname.
path	The path.

1.23.2.4 enwrap_with_outer_url()

```
def shallot.Eurl.enwrap_with_outer_url (
              self,
              scheme,
              hostname,
              path )
```

Returns a new shallot.Eurl containing this one packed as embedding and new outer parts `scheme`, `hostname` and `path`.

Example: `scheme:/[file:///a/b/c.zip]/hostname/p/a/t/h` for `file:///a/b/c.zip`.

Parameters

scheme	The scheme part of the new outer eurl.
hostname	The hostname of the new outer eurl.
path	The path of the new outer eurl.

1.23.2.5 from_string()

```
def shallot.Eurl.from_string (
              s )  [static]
```

Returns a shallot.Eurl from a string.

Parameters

s	The eurl string. This is what shallot.Eurl.as_string would return.

1.23.2.6 has_inner_urls()

```
def shallot.Eurl.has_inner_urls (
              self )
```

Checks if this shallot.Eurl has embeddings.

Examples: `true` for `foo:/[bar:///foo]/host/`. `false` for `foo://host/`.

1.23.2.7 has_parent_segment()

```
def shallot.Eurl.has_parent_segment (
              self )
```

Checks if this shallot.Eurl has a parent segment.

This indicates if shallot.Eurl.parent_segment would return `0`.

1.23.2.8 hostname()

```
def shallot.Eurl.hostname (
              self )
```

Returns the hostname part (from the outer url of this shallot.Eurl).

Examples: `"livingroom-pc"` for `smb://livingroom-pc/foo/bar/baz`. `""` for `file:///foo/bar/baz`.

1.23.2.9 is_prefix_of()

```
def shallot.Eurl.is_prefix_of (
              self,
              eurl )
```

Checks if this shallot.Eurl is a prefix of another one.

This is not an equivalent to a string comparison but it checks parent relationships according to shallot.Eurl.parent_segment.

Parameters

eurl	The longer shallot.Eurl.

1.23.2.10 outer_url()

```
def shallot.Eurl.outer_url (
              self )
```

Returns a new shallot.Eurl containing only the outer part of this one (strips the embeddings).

Example: `foobar://host/foo/bar/baz` for `foobar:/[zip:/[file:///a/b/c.zip]//d/e]//foo/bar/baz`.

1.23.2.11 outer_url_is_root_directory()

```
def shallot.Eurl.outer_url_is_root_directory (
              self )
```

Checks if this shallot.Eurl is a root path (with or without embeddings).

Examples: `True` for `foo://host/`. `False` for `foo://host/a`. `True` for `foo:/[bar:///goo]/host/`. `False` for `foo:/[bar:///goo]/host/a`. `True` for `foo:/[bar:///goo]//`. `False` for `foo:/[bar:///goo]//a`.

1.23.2.12 outermost_inner_eurl()

```
def shallot.Eurl.outermost_inner_eurl (
            self )
```

Returns a new shallot.Eurl containing only the embedding of this one.

Example: `zip:/[file:///a/b/c.zip]//d/e` for `foobar:/[zip:/[file:///a/b/c.zip]//d/e]//foo/bar/b`

1.23.2.13 parent_segment()

```
def shallot.Eurl.parent_segment (
            self )
```

Returns the parent shallot.Eurl.

At first, this traverses path segments. For a root path eurl with embeddings, it returns the embedding. If none are available, it returns `0`. Examples: `foo://host/foo` for `foo://host/foo/bar`. `foo://host/` for `foo://host/boo`. `foo:/[bar://host/goo]/anotherhost/` for `foo:/[bar://host/goo]/anotherhost/boo`. `bar://host/foo` for `foo:/[bar://host/foo]/host/`. `0` for `foo://host/`.

1.23.2.14 path()

```
def shallot.Eurl.path (
            self )
```

Returns the path part (from the outer url of this shallot.Eurl).

Examples: `"/foo/bar/baz"` for `smb://livingroom-pc/foo/bar/baz`. `"/foo/bar/baz"` for `file:///foo/bar/baz`. `"/"` for `file:///`.

1.23.2.15 root()

```
def shallot.Eurl.root (
            self )
```

Returns the root shallot.Eurl from this one.

Example: `zip:/[file:///a/b/c.zip]//` for `zip:/[file:///a/b/c.zip]//foo/bar/baz`.

1.23.2.16 scheme()

```
def shallot.Eurl.scheme (
            self )
```

Returns the scheme (from the outer url of this shallot.Eurl).

This is what comes before the `://`. Example: `"file"` for `file:///foo/bar/baz`.

1.23.2.17 with_appended_segment()

```
def shallot.Eurl.with_appended_segment (
            self,
            segment )
```

Returns a new shallot.Eurl from this one with "/basename" appended.

The parameter is assumed to be a single path segment.

Parameters

segment	The basename to be appended.

1.23.2.18 with_appended_segments()

```
def shallot.Eurl.with_appended_segments (
              self,
              path )
```

Returns a new shallot.Eurl with path segments "/pa/t/h/" appended.

The parameter may contain "/"s for dividing path segments.

Parameters

path	The path segments to be appended, like "`foo/bar`".

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

- /home/pino/projects/shallot/shallot.py

1.24 shallot.Exceptions Class Reference

Everything about exceptions.

Classes

- class ArgumentException

  Shallot exception for failed operation due to invalid arguments given to some program part.
- class IOException

  Shallot exception in IO.
- class ProgramException

  Shallot exception for program errors (typically logical stuff) in the program or plugin.
- class RuntimeException

  Shallot exception for failed operation due to (often external) runtime effects.
- class ScriptedException

  The Shallot exception class.

1.24.1 Detailed Description

Everything about exceptions.

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

- /home/pino/projects/shallot/shallot.py

1.25 shallot.Actions.ExecutionInfo Class Reference

An object for signalling action execution state changes between an action implementation and the Shallot core (mostly to the core).

Public Member Functions

- def __init__ (self)
  Can't be constructed directly.
- def set_visualprocessfeedback_active (self, v)
  Sets the visibility of a visual feedback.
- def is_visualprocessfeedback_active (self)
  Is visual feedback visibility enforced?
- def set_manual_intervention_needed (self, v)
  Set if manual intervention is needed.
- def is_manual_intervention_needed (self)
  Is manual intervention needed?
- def set_details (self, fromverb, fromobjectname, toverb, toobjectname)
  Sets the progress details.
- def from_verb (self)
  Gets the current verb on from-side.
- def from_objectname (self)
  Gets the current object name on from-side.
- def to_verb (self)
  Gets the current verb on to-side.
- def to_objectname (self)
  Gets the current object name on to-side.
- def set_head (self, txt)
  Sets the head text.
- def head (self)
  Gets the head text.
- def set_progress (self, done, all, label)
  Sets a current determinate progress.
- def set_progress_indeterminate (self, label)
  Sets a current indeterminate progress.
- def progress_done (self)
  How many items are done?
- def progress_all (self)
  How many items are to do in sum?
- def progress_text (self)
  The textual representation of the progress.
- def is_cancelled (self)
  Is the action cancelled?
- def cancel (self)
  Cancel the action.
- def respect_cancel (self)
  Respect a cancel request.
- def operation (self)
  Gets the shallot.Operations.Operation transactional operation object.
- def add_changed_eurl (self, eurl)
  Requests updating the filesystem model info for an item.
- def userfeedback (self)
  The shallot.Actions.ExecutionUserFeedback user feedback object.

1.25.1 Detailed Description

An object for signalling action execution state changes between an action implementation and the Shallot core (mostly to the core).

Most calls lead to changes in the information presented by the progress dialog.

1.25.2 Member Function Documentation

1.25.2.1 add_changed_eurl()

```
def shallot.Actions.ExecutionInfo.add_changed_eurl (
              self,
              eurl )
```

Requests updating the filesystem model info for an item.

Parameters

eurl	The item location to be updated as shallot.Eurl.

1.25.2.2 respect_cancel()

```
def shallot.Actions.ExecutionInfo.respect_cancel (
              self )
```

Respect a cancel request.

This should be called from time to time (inside loops for example).

1.25.2.3 set_details()

```
def shallot.Actions.ExecutionInfo.set_details (
              self,
              fromverb,
              fromobjectname,
              toverb,
              toobjectname )
```

Sets the progress details.

Parameters

fromverb	The verb on from-side (the source).
fromobjectname	(file path) The objectname on from-side (the source).
toverb	The verb on to-side (the destination).
toobjectname	(file path) The objectname on to-side (the destination).

1.25.2.4 set_progress()

```
def shallot.Actions.ExecutionInfo.set_progress (
              self,
              done,
              all,
              label )
```

Sets a current determinate progress.

Parameters

done	The number of finished items.
all	The gross number of items.
label	Progress description text.

1.25.2.5 set_progress_indeterminate()

```
def shallot.Actions.ExecutionInfo.set_progress_indeterminate (
              self,
              label )
```

Sets a current indeterminate progress.

Parameters

label	Progress description text.

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

- /home/pino/projects/shallot/shallot.py

1.26 shallot.Actions.ExecutionUserFeedback Class Reference

An object for communication with the user in action implementations.

Classes

- class MessageBoxButton

  Buttons in a message box from shallot.Actions.ExecutionUserFeedback.

Public Member Functions

- def __init__ (self)

  Can't be constructed directly.
- def simple_messagebox (self, message, buttons, icon, defaultbutton, abortbutton)

  A simple message box.
- def messagebox (self, message, buttons, icon, defaultbutton, abortbutton)

  A message box.
- def simple_inputbox (self, question, defaulttext)

  A simple input box.

1.26.1 Detailed Description

An object for communication with the user in action implementations.

It can be used to ask the user for some information or to just give some information to the user.

1.26.2 Member Function Documentation

1.26.2.1 messagebox()

```
def shallot.Actions.ExecutionUserFeedback.messagebox (
              self,
              message,
              buttons,
              icon,
              defaultbutton,
              abortbutton )
```

A message box.

Returns the index of the chosen button.

Parameters

message	The message text.
buttons	A list of buttons as string list.
icon	An icon name.
defaultbutton	Answer for keyboard Enter as shallot.Actions.ExecutionUserFeedback.MessageBoxButton.
abortbutton	Answer for keyboard Esc as shallot.Actions.ExecutionUserFeedback.MessageBoxButton.

1.26.2.2 simple_inputbox()

```
def shallot.Actions.ExecutionUserFeedback.simple_inputbox (
              self,
```

```
                question,
                defaulttext )
```

A simple input box.

Parameters

question	The question.
defaulttext	The default answer.

1.26.2.3 simple_messagebox()

```
def shallot.Actions.ExecutionUserFeedback.simple_messagebox (
                self,
                message,
                buttons,
                icon,
                defaultbutton,
                abortbutton )
```

A simple message box.

Returns the chosen button.

Parameters

message	The message text.
buttons	A list of buttons as or-combination of shallot.Actions.ExecutionUserFeedback.MessageBoxButton.
icon	An icon name.
defaultbutton	Answer for keyboard Enter as shallot.Actions.ExecutionUserFeedback.MessageBoxButton. May be 0 for no definition.
abortbutton	Answer for keyboard Esc as shallot.Actions.ExecutionUserFeedback.MessageBoxButton. May be 0 for no definition.

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

- /home/pino/projects/shallot/shallot.py

1.27 shallot.FilePropertyDialog Class Reference

The file property dialog.

Classes

- class Tab

  Abstract base class for a tab in the Properties dialog.
- class TabPropertyIconTextBanner

  Represents values for image/text banners as used for shallot.FilePropertyDialog.Tab.PropertyType.IconTextBanner.

Public Attributes

- **tabclass**

1.27.1 Detailed Description

The file property dialog.

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

- /home/pino/projects/shallot/shallot.py

1.28 shallot.FileSearch Class Reference

File searches.

Classes

- class SearchCriterion

  Abstract base class for a search criterion.
- class SearchCriterionFactory

  A factory for a shallot.FileSearch.SearchCriterion class.

1.28.1 Detailed Description

File searches.

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

- /home/pino/projects/shallot/shallot.py

1.29 shallot.Filesystem Class Reference

The filesystem.

Classes

- class Handler

  Abstract base class for a custom filesystem handler.
- class Node

  A model representation of a location in the filesystem tree.
- class NodeList

  A list editor for a filesystem node list.
- class NodeType

  Enumeration of types a shallot.Filesystem.Node can have.

Static Public Member Functions

- def find_nodes_for_eurl (eurl)

  Get a list of shallot.Filesystem.Node for a shallot.Eurl.
- def try_get_nodes_for_eurl (eurl)

  Returns a list of shallot.Filesystem.Node for a shallot.Eurl.
- def refresh (eurl, forceFindParent=False, withDetails=True)

  Request to refresh the internal model information for a shallot.Eurl.
- def get_or_create_node (eurl, handler, nodetype, parentnode, doinsert=True, showInitialLoadingLabel=True, hidden=False)

  Returns a shallot.Filesystem.Node for using it as a child node in the model.
- def create_node (eurl, handler, nodetype, parentnode, doinsert=True, showInitialLoadingLabel=True, hidden=False)

  Creates a new shallot.Filesystem.Node for placing it into the model.

Static Public Attributes

- rootnode = internalstuff.modelRootNode()

  The shallot.Filesystem.Node which is the root node of the entire model.

1.29.1 Detailed Description

The filesystem.

1.29.2 Member Function Documentation

1.29.2.1 create_node()

```
def shallot.Filesystem.create_node (
              eurl,
              handler,
              nodetype,
              parentnode,
              doinsert = True,
              showInitialLoadingLabel = True,
              hidden = False )  [static]
```

Creates a new shallot.Filesystem.Node for placing it into the model.

If such a node (with the same eurl for the same parent node) does not exist, it generates a new one. If there already is such a node alive, but currently not placed in the model, it recycles this one. This can happen when references exist to a node which is not yet inserted or which is removed meanwhile. It is not allowed to call this method when such a node already exists in the model. Use this function for getting a shallot.Filesystem.Node, which is to be added to the model now or later. It is typically used within a shallot.Filesystem.Handler implementation.

Parameters

eurl	The shallot.Eurl of the new item.
handler	The shallot.Filesystem.Handler of the new node. This must always be the same one as the handler registered for the scheme of the eurl!
nodetype	The shallot.Filesystem.NodeType node type.
parentnode	The shallot.Filesystem.Node parent node.
doinsert	If this function shall actually add the node to the model (to be exact, in some situations, this will not actually take place nonetheless).
showInitialLoadingLabel	If a 'loading...' label shall be visible at beginning.
hidden	If the node shall be created as hidden one. See also shallot.Filesystem.Node.ishidden.

1.29.2.2 find_nodes_for_eurl()

```
def shallot.Filesystem.find_nodes_for_eurl (
              eurl )  [static]
```

Get a list of shallot.Filesystem.Node for a shallot.Eurl.

If it is unknown to the model so far, it tries to build them. In typical cases, this list either contains one element, or is empty if the filesystem handlers decide that this file does not exist. But in some cases, there is also more than one node for one shallot.Eurl (when the Eurl appears on more than one place in the tree). It only returns nodes, which are 'alive', i.e. which have a living parent and which are a child of this parent.

1.29.2.3 get_or_create_node()

```
def shallot.Filesystem.get_or_create_node (
              eurl,
              handler,
              nodetype,
              parentnode,
              doinsert = True,
              showInitialLoadingLabel = True,
              hidden = False )  [static]
```

Returns a shallot.Filesystem.Node for using it as a child node in the model.

It either creates a new one, if there currently is no node for this eurl in this parentnode, or returns the existing one. Even for existing ones, this call can change the nodetype of that node.

Parameters

eurl	The shallot.Eurl of the new node.
handler	The shallot.Filesystem.Handler of the new node. This must always be the same one as the handler registered for the scheme of the eurl!
nodetype	The shallot.Filesystem.NodeType node type.
parentnode	The shallot.Filesystem.Node parent node.
doinsert	If this function shall actually add the node to the model (to be exact, in some situations, this will not actually take place nonetheless).
showInitialLoadingLabel	If a 'loading...' label shall be visible at beginning.
hidden	If the node shall be created as hidden one. See also shallot.Filesystem.Node.ishidden.

1.29.2.4 refresh()

```
def shallot.Filesystem.refresh (
              eurl,
              forceFindParent = False,
              withDetails = True )  [static]
```

Request to refresh the internal model information for a shallot.Eurl.

This may be a place which is already known (then a change of some metadata or the removal is detected) or a formerly unknown place (then new nodes get inserted in the model).

Parameters

eurl	The shallot.Eurl of the node.
forceFindParent	Forcefully create such model information if unknown before.
withDetails	If detail columns assigned to an existing node should also be updated.

1.29.2.5 try_get_nodes_for_eurl()

```
def shallot.Filesystem.try_get_nodes_for_eurl (
              eurl )  [static]
```

Returns a list of shallot.Filesystem.Node for a shallot.Eurl.

It only considers the current state of the in-memory model. It will only return nodes which are already known to the model so far. This operation is cheaper and handling only the known nodes is sufficient in many situations. In typical cases, this list either contains one element, or is empty. But in some cases, there is also more than one node for one shallot.Eurl (when the Eurl appears on more than one place in the tree). It only returns nodes, which are 'alive', i.e. which have a living parent and which are a child of this parent.

1.29.3 Member Data Documentation

1.29.3.1 rootnode

```
shallot.Filesystem.rootnode = internalstuff.modelRootNode()  [static]
```

The shallot.Filesystem.Node which is the root node of the entire model.

It is the parent for all toplevel nodes.

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

- /home/pino/projects/shallot/shallot.py

1.30 shallot.Operations.FilesystemOperation Class Reference

A high-level provider of filesystem operations.

Public Member Functions

- def __init__ (self)

  Can't be constructed directly.
- def itemlist (self, eurl)

  Gets a list of shallot.Eurl in a directory.
- def itemlist_by_type (self, eurl, nodetype)

  Gets a list of shallot.Eurl in a directory which have a certain type.
- def get_type (self, eurl)

  Gets the shallot.Filesystem.NodeType node type for an entry.
- def get_filesize (self, eurl)

  Gets the file size in bytes for an entry.
- def get_linktarget (self, eurl)

  Gets the link target for an entry (if it is a link).
- def can_get_filecontent (self, eurl)

  Can we get the file content for an entry?
- def get_file_content (self, eurl)

  Gets the file content for an entry as shallot.Streaming.ReadDataDevice.
- def can_create_directory (self, eurl)

  Can we create a given directory?
- def create_directory (self, eurl, progressmon)

  Creates a directory.
- def can_create_link (self, eurl)

  Can we create a given link?
- def create_link (self, eurl, tgt, progressmon)

  Creates a link.
- def can_create_file (self, eurl)

  Can we create a given file?
- def create_file (self, eurl, contentdevice, progressmon)

  Create a file.
- def can_delete_item (self, eurl)

  Can we delete a given entry?
- def delete_item (self, eurl, progressmon)

  Delete an entry.
- def delete_directory_if_empty (self, eurl, progressmon)

  Delete a directory entry if empty.
- def can_move_item (self, src, dest)

  Can we move a given entry?
- def move_items (self, src, tgt, progressmon)

  Move entries.
- def move_item (self, src, tgt, progressmon)

  Move an entry.
- def can_copy_item (self, src, dest)

  Can we copy a given entry?
- def copy_items (self, src, tgt, progressmon)

Copy entries.

- def copy_item (self, src, tgt, progressmon)

  Copy an entry.
- def list_extendedattributes (self, eurl)

  Lists all keys of extended attributes stored in the filesystem for an entry.
- def get_extendedattribute_size (self, eurl, attribute)

  Returns the size of an extended attribute value stored in the filesystem for an entry.
- def get_extendedattribute (self, eurl, attribute)

  Returns the value (as byte string) of an extended attribute value stored in the filesystem for an entry.
- def set_extendedattribute (self, eurl, attribute, value)

  Store the value of an extended attribute in the filesystem for an entry.
- def remove_extendedattribute (self, eurl, attribute)

  Removes an extended attribute stored in the filesystem for an entry.
- def resolve_node_link (self, node)

  Resolve a link as node.
- def resolve_node_link_nonrecursive (self, node)

  Resolve a link as node without recursion.
- def resolve_eurl_link (self, eurl)

  Resolve a link as Eurl.
- def resolve_eurl_link_nonrecursive (self, eurl)

  Resolve a link as Eurl without recursion.

1.30.1 Detailed Description

A high-level provider of filesystem operations.

It is always based on the transaction of a shallot.Operations.Operation and also accessible from such an instance.

Some operations allow to specify an shallot.Operations.FilesystemOperationProgressMonitor. You can specify an instance for getting some additional functionaly, like progress notification, conflict resolution and more. Please note that not each operation uses each aspect of shallot.Operations.FilesystemOperationProgressMonitor (e.g. some will not do any conflict resolution).

1.30.2 Member Function Documentation

1.30.2.1 can_copy_item()

```
def shallot.Operations.FilesystemOperation.can_copy_item (
            self,
            src,
            dest )
```

Can we copy a given entry?

Parameters

src	The shallot.Eurl of the item to be copied.
dest	The shallot.Eurl destination path. If it is not known, use `None`.

1.30.2.2 can_create_directory()

```
def shallot.Operations.FilesystemOperation.can_create_directory (
              self,
              eurl )
```

Can we create a given directory?

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.3 can_create_file()

```
def shallot.Operations.FilesystemOperation.can_create_file (
              self,
              eurl )
```

Can we create a given file?

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.4 can_create_link()

```
def shallot.Operations.FilesystemOperation.can_create_link (
              self,
              eurl )
```

Can we create a given link?

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.5 can_delete_item()

```
def shallot.Operations.FilesystemOperation.can_delete_item (
```

```
                self,
                eurl )
```

Can we delete a given entry?

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.6 can_get_filecontent()

```
def shallot.Operations.FilesystemOperation.can_get_filecontent (
                self,
                eurl )
```

Can we get the file content for an entry?

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.7 can_move_item()

```
def shallot.Operations.FilesystemOperation.can_move_item (
                self,
                src,
                dest )
```

Can we move a given entry?

Parameters

src	The shallot.Eurl of the item to be moved.
dest	The shallot.Eurl destination path. If it is not known, use `None`.

1.30.2.8 copy_item()

```
def shallot.Operations.FilesystemOperation.copy_item (
                self,
                src,
                tgt,
                progressmon )
```

Copy an entry.

Parameters

src	The shallot.Eurl of the item to be copied.
tgt	The shallot.Eurl of the new destination.
progressmon	A shallot.Operations.FilesystemOperationProgressMonitor instance for some additional functionality (or `None`). See shallot.Operations.FilesystemOperation for details.

1.30.2.9 copy_items()

```
def shallot.Operations.FilesystemOperation.copy_items (
              self,
              src,
              tgt,
              progressmon )
```

Copy entries.

Parameters

src	List of shallot.Eurl of the item to be copied.
tgt	The shallot.Eurl of the new common parent destination.
progressmon	A shallot.Operations.FilesystemOperationProgressMonitor instance for some additional functionality (or `None`). See shallot.Operations.FilesystemOperation for details.

1.30.2.10 create_directory()

```
def shallot.Operations.FilesystemOperation.create_directory (
              self,
              eurl,
              progressmon )
```

Creates a directory.

Parameters

eurl	The shallot.Eurl of the new directory.
progressmon	A shallot.Operations.FilesystemOperationProgressMonitor instance for some additional functionality (or `None`). See shallot.Operations.FilesystemOperation for details.

1.30.2.11 create_file()

```
def shallot.Operations.FilesystemOperation.create_file (
              self,
```

```
                eurl,
                contentdevice,
                progressmon )
```

Create a file.

Parameters

eurl	The shallot.Eurl of the new file.
contentdevice	a shallot.Streaming.ReadDataDevice content device.
progressmon	A shallot.Operations.FilesystemOperationProgressMonitor instance for some additional functionality (or `None`). See shallot.Operations.FilesystemOperation for details.

1.30.2.12 create_link()

```
def shallot.Operations.FilesystemOperation.create_link (
                self,
                eurl,
                tgt,
                progressmon )
```

Creates a link.

Parameters

eurl	The shallot.Eurl of the new link.
tgt	Target path of this link.
progressmon	A shallot.Operations.FilesystemOperationProgressMonitor instance for some additional functionality (or `None`). See shallot.Operations.FilesystemOperation for details.

1.30.2.13 delete_directory_if_empty()

```
def shallot.Operations.FilesystemOperation.delete_directory_if_empty (
                self,
                eurl,
                progressmon )
```

Delete a directory entry if empty.

Parameters

eurl	The shallot.Eurl of the directory to delete.
progressmon	A shallot.Operations.FilesystemOperationProgressMonitor instance for some additional functionality (or `None`). See shallot.Operations.FilesystemOperation for details.

1.30.2.14 delete_item()

```
def shallot.Operations.FilesystemOperation.delete_item (
              self,
              eurl,
              progressmon )
```

Delete an entry.

Parameters

eurl	The shallot.Eurl of the item to delete.
progressmon	A shallot.Operations.FilesystemOperationProgressMonitor instance for some additional functionality (or `None`). See shallot.Operations.FilesystemOperation for details.

1.30.2.15 get_extendedattribute()

```
def shallot.Operations.FilesystemOperation.get_extendedattribute (
              self,
              eurl,
              attribute )
```

Returns the value (as byte string) of an extended attribute value stored in the filesystem for an entry.

Parameters

eurl	The shallot.Eurl of the considered item.
attribute	The attribute key.

1.30.2.16 get_extendedattribute_size()

```
def shallot.Operations.FilesystemOperation.get_extendedattribute_size (
              self,
              eurl,
              attribute )
```

Returns the size of an extended attribute value stored in the filesystem for an entry.

Parameters

eurl	The shallot.Eurl of the considered item.
attribute	The attribute key.

1.30.2.17 get_file_content()

```
def shallot.Operations.FilesystemOperation.get_file_content (
              self,
              eurl )
```

Gets the file content for an entry as shallot.Streaming.ReadDataDevice.

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.18 get_filesize()

```
def shallot.Operations.FilesystemOperation.get_filesize (
              self,
              eurl )
```

Gets the file size in bytes for an entry.

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.19 get_linktarget()

```
def shallot.Operations.FilesystemOperation.get_linktarget (
              self,
              eurl )
```

Gets the link target for an entry (if it is a link).

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.20 get_type()

```
def shallot.Operations.FilesystemOperation.get_type (
              self,
              eurl )
```

Gets the shallot.Filesystem.NodeType node type for an entry.

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.21 itemlist()

```
def shallot.Operations.FilesystemOperation.itemlist (
              self,
              eurl )
```

Gets a list of shallot.Eurl in a directory.

Parameters

eurl	The shallot.Eurl of the directory to list.

1.30.2.22 itemlist_by_type()

```
def shallot.Operations.FilesystemOperation.itemlist_by_type (
              self,
              eurl,
              nodetype )
```

Gets a list of shallot.Eurl in a directory which have a certain type.

Parameters

eurl	The shallot.Eurl of the directory to list.
nodetype	The shallot.Filesystem.NodeType node type to fetch.

1.30.2.23 list_extendedattributes()

```
def shallot.Operations.FilesystemOperation.list_extendedattributes (
              self,
              eurl )
```

Lists all keys of extended attributes stored in the filesystem for an entry.

Parameters

eurl	The shallot.Eurl of the considered item.

1.30.2.24 move_item()

```
def shallot.Operations.FilesystemOperation.move_item (
              self,
              src,
              tgt,
              progressmon )
```

Move an entry.

Parameters

src	The shallot.Eurl of the item to be moved.
tgt	The shallot.Eurl of the new destination.
progressmon	A shallot.Operations.FilesystemOperationProgressMonitor instance for some additional functionality (or `None`). See shallot.Operations.FilesystemOperation for details.

1.30.2.25 move_items()

```
def shallot.Operations.FilesystemOperation.move_items (
              self,
              src,
              tgt,
              progressmon )
```

Move entries.

Parameters

src	List of shallot.Eurl of the item to be moved.
tgt	The shallot.Eurl of the new common parent destination.
progressmon	A shallot.Operations.FilesystemOperationProgressMonitor instance for some additional functionality (or `None`). See shallot.Operations.FilesystemOperation for details.

1.30.2.26 remove_extendedattribute()

```
def shallot.Operations.FilesystemOperation.remove_extendedattribute (
              self,
              eurl,
              attribute )
```

Removes an extended attribute stored in the filesystem for an entry.

Parameters

eurl	The shallot.Eurl of the considered item.
attribute	The attribute key.

1.30.2.27 resolve_eurl_link()

```
def shallot.Operations.FilesystemOperation.resolve_eurl_link (
              self,
              eurl )
```

Resolve a link as Eurl.

Parameters

eurl	A shallot.Eurl to resolve.

1.30.2.28 resolve_eurl_link_nonrecursive()

```
def shallot.Operations.FilesystemOperation.resolve_eurl_link_nonrecursive (
              self,
              eurl )
```

Resolve a link as Eurl without recursion.

Parameters

eurl	A shallot.Eurl to resolve.

1.30.2.29 resolve_node_link()

```
def shallot.Operations.FilesystemOperation.resolve_node_link (
              self,
              node )
```

Resolve a link as node.

Parameters

node	A shallot.Filesystem.Node to resolve.

1.30.2.30 resolve_node_link_nonrecursive()

```
def shallot.Operations.FilesystemOperation.resolve_node_link_nonrecursive (
              self,
              node )
```

Resolve a link as node without recursion.

Parameters

node	A shallot.Filesystem.Node to resolve.

1.30.2.31 set_extendedattribute()

```
def shallot.Operations.FilesystemOperation.set_extendedattribute (
              self,
              eurl,
              attribute,
              value )
```

Store the value of an extended attribute in the filesystem for an entry.

Parameters

eurl	The shallot.Eurl of the considered item.
attribute	The attribute key.
value	The attribute value (as byte string).

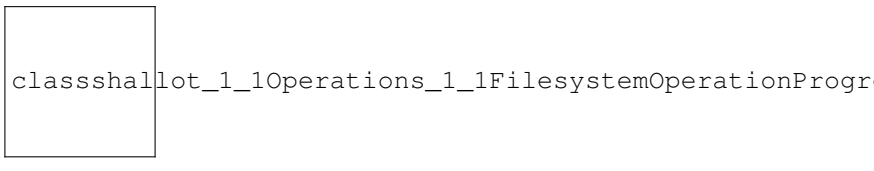
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.31 shallot.Operations.FilesystemOperationProgressMonitor Class Reference

Abstract base class for progress monitors, used for monitoring progress of some operations in shallot.Operations.↩ FilesystemOperation and for conflict resolution.

Inheritance diagram for shallot.Operations.FilesystemOperationProgressMonitor:



Public Member Functions

- def __init__ (self, actionExecution=None)
- def changed (self)

  Reacts on progress changes.
- def resolve_conflicts (self, steps)

  Resolves upcoming filesystem conflicts.
- def has_item_info (self)

  If the monitor currently has information about progress in terms of item counts.
- def done_items (self)

  How many items are transferred so far.
- def all_items (self)

  How many items are to be transferred in total.
- def has_bytes_info (self)

  If the monitor currently has information about progress in terms of transferred bytes.
- def done_bytes (self)

  How many bytes are transferred so far.
- def all_bytes (self)

  How many bytes are to be transferred in total.
- def get_item_info_from (self)

  The current source as textual representation.
- def get_item_info_to (self)

  The current destination as textual representation.
- def estimation (self)

  The current performance estimation as textual representation.

1.31.1 Detailed Description

Abstract base class for progress monitors, used for monitoring progress of some operations in shallot.Operations.↩ FilesystemOperation and for conflict resolution.

It makes sense to directly instantiate this class in some cases.

1.31.2 Constructor & Destructor Documentation

1.31.2.1 __init__()

```
def shallot.Operations.FilesystemOperationProgressMonitor.__init__ (
              self,
              actionExecution = None )
```

Parameters

actionExecution	Optional instance of shallot.Actions.ExecutionInfo. If available, it provides some additional features (e.g. the user may cancel the transfer).

1.31.3 Member Function Documentation

1.31.3.1 changed()

```
def shallot.Operations.FilesystemOperationProgressMonitor.changed (
              self )
```

Reacts on progress changes.

Override this method in custom subclasses or leave the default implementation.

1.31.3.2 resolve_conflicts()

```
def shallot.Operations.FilesystemOperationProgressMonitor.resolve_conflicts (
              self,
              steps )
```

Resolves upcoming filesystem conflicts.

Override this method in custom subclasses or leave the default implementation.

Parameters

steps	A list of shallot.Operations.FilesystemOperationStep which stay in conflict.

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

- /home/pino/projects/shallot/shallot.py

1.32 shallot.Operations.FilesystemOperationStep Class Reference

One step in a larger filesystem transfer running inside some operations in shallot.Operations.FilesystemOperation.

Classes

- class ConflictResolution

  Enumeration of conflict resolution strategies.

Public Member Functions

- def __init__ (self)

  Can't be constructed directly.
- def conflict_description (self)

  The conflict descrition text.
- def conflict_resolution_rename_destination_before_to (self)

  The new destination name (if conflict_resolution is shallot.Operations.FilesystemOperationStep.ConflictResolution.RenameDestinationBefore)
- def conflict_resolution_different_destination_name_to (self)

  The new destination name (if conflict_resolution is shallot.Operations.FilesystemOperationStep.ConflictResolution.UseDifferentDestinationName)
- def set_conflict_resolve_skip (self)

  Set the conflict resolution to shallot.Operations.FilesystemOperationStep.ConflictResolution.Skip.
- def set_conflict_resolve_overwrite_destination (self)

  Set the conflict resolution to shallot.Operations.FilesystemOperationStep.ConflictResolution.OverwriteDestination.
- def set_conflict_resolve_rename_destination_before (self, newname)

  Set the conflict resolution to shallot.Operations.FilesystemOperationStep.ConflictResolution.RenameDestinationBefore.
- def set_conflict_resolve_use_different_destination_name (self, newname)

  Set the conflict resolution to shallot.Operations.FilesystemOperationStep.ConflictResolution.UseDifferentDestinationName.
- def set_conflict_resolve_merge_directories (self)

  Set the conflict resolution to shallot.Operations.FilesystemOperationStep.ConflictResolution.MergeDirectories.
- def sourcetype (self)

  The shallot.Filesystem.NodeType of the source.
- def source (self)

  The source shallot.Eurl location.
- def original_destination (self)

  The original destination shallot.Eurl location.
- def effective_destination (self)

  The effective destination shallot.Eurl location with applied conflict resolution strategies.
- def set_conflict_resolve_merge_directories_check_allowed (self)

  Checks if it is allowed to solve this conflict by merging directories.
- def conflict_resolution (self)

  The currently selected shallot.Operations.FilesystemOperationStep.ConflictResolution conflict resolution.

1.32.1 Detailed Description

One step in a larger filesystem transfer running inside some operations in shallot.Operations.FilesystemOperation.

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

- /home/pino/projects/shallot/shallot.py

1.33 shallot.Setting.GroupInfo Class Reference

Enumeration of groups to which a setting can belong.

Static Public Attributes

- Global = internalstuff.settinggroup_global()
  Global aspects group.
- GUI = internalstuff.settinggroup_gui()
  User interface aspects group.
- FileHandling = internalstuff.settinggroup_filehandling()
  File handling aspects group.
- DataView = internalstuff.settinggroup_dataview()
  Data view aspects group.
- Behavior = internalstuff.settinggroup_behavior()
  Behavioral aspects group.
- Special = internalstuff.settinggroup_special()
  Special aspects group.

1.33.1 Detailed Description

Enumeration of groups to which a setting can belong.

This is just a matter of grouping the for presentation.

1.33.2 Member Data Documentation

1.33.2.1 Behavior

```
shallot.Setting.GroupInfo.Behavior = internalstuff.settinggroup_behavior()  [static]
```

Behavioral aspects group.

1.33.2.2 DataView

```
shallot.Setting.GroupInfo.DataView = internalstuff.settinggroup_dataview()  [static]
```

Data view aspects group.

1.33.2.3 FileHandling

```
shallot.Setting.GroupInfo.FileHandling = internalstuff.settinggroup_filehandling()  [static]
```

File handling aspects group.

1.33.2.4 Global

```
shallot.Setting.GroupInfo.Global = internalstuff.settinggroup_global()  [static]
```

Global aspects group.

1.33.2.5 GUI

```
shallot.Setting.GroupInfo.GUI = internalstuff.settinggroup_gui()  [static]
```

User interface aspects group.

1.33.2.6 Special

```
shallot.Setting.GroupInfo.Special = internalstuff.settinggroup_special()  [static]
```

Special aspects group.

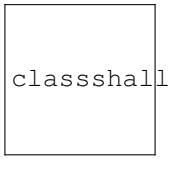
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.34 shallot.Filesystem.Handler Class Reference

Abstract base class for a custom filesystem handler.

Inheritance diagram for shallot.Filesystem.Handler:



Public Member Functions

- def **__init__** (self)
- def configure_item (self, operation, node)

  Configure a newly created node (e.g.
- def list_extendedattributes (self, operation, eurl)

  Returns a list of extended attributes which exist for a location.
- def get_extendedattribute_size (self, operation, eurl, attribname)

  Returns the size (in byte) of the value for a particular extended attribute.
- def get_extendedattribute (self, operation, eurl, attribname)

  Gets the value for a particular extended attribute.
- def set_extendedattribute (self, operation, eurl, attribname, value)

  Sets the value for a particular extended attribute.
- def remove_extendedattribute (self, operation, eurl, attribname)

  Removes an extended attribute.
- def get_customattributes (self, operation, eurl)

  Returns a dict<string,string> with custom attributes for a node.
- def set_customattribute (self, operation, eurl, key, value)

  Sets a custom attribute to an entry.
- def get_mimetype (self, operation, eurl)

  Determine mime type of a file.
- def get_type (self, operation, eurl)

  Determine if a file is regular, or a dir, or a link, ...
- def get_size (self, operation, eurl)

  Determine the size of a file.
- def get_mtime (self, operation, eurl)

  Determine the mtime of a file.
- def set_mtime (self, operation, eurl, mtime)

  Set the mtime of a file.
- def can_create_directory (self, operation, eurl)

  Is it allowed to create subdirectories in a certain directory?
- def create_directory (self, operation, eurl)

  Create a directory.
- def can_create_file (self, operation, eurl)

  Is it allowed to create files in a certain directory?
- def can_delete_item (self, operation, eurl)

  Is it allowed to delete a certain file/directory/link/...?
- def delete_item (self, operation, eurl)

  Delete a file/directory/link/...
- def can_rename_item (self, operation, src)

  Is it allowed to move a certain file/directory/link/...?
- def rename_item (self, operation, src, destpath)

  Move a file/directory/link/...
- def itemlist (self, operation, eurl, nodetype, nodelist)

  Determine a list of subelements in a certain directory.
- def can_get_filecontent (self, operation, eurl)

  Is it allowed to get the content of a certain file?
- def get_file_content (self, operation, eurl)

  Get the content of a file as a shallot.Streaming.ReadDataDevice.
- def create_file (self, operation, eurl, outstream, handlertransfer)

  Creates a file with some content.

- def get_actions (self, eurls)

  Which actions (see former example) are to be offered for certain files?
- def can_create_link (self, operation, eurl)

  Is it allowed to create a link in a certain directory?
- def create_link (self, operation, eurl, tgt)

  Create a link.
- def get_linktarget (self, operation, eurl)

  Resolve a link.

Static Public Member Functions

- def register (scheme, handler)

  Registers a shallot.Filesystem.Handler filesystem handler implementation.

1.34.1 Detailed Description

Abstract base class for a custom filesystem handler.

Subclass and register it for implementing a new virtual filesystem, which lets new nodes appear somewhere in the filesystem tree and controls how to handle them. Use shallot.Filesystem.Handler.register for registration.

1.34.2 Member Function Documentation

1.34.2.1 can_create_directory()

```
def shallot.Filesystem.Handler.can_create_directory (
              self,
              operation,
              eurl )
```

Is it allowed to create subdirectories in a certain directory?

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.2 can_create_file()

```
def shallot.Filesystem.Handler.can_create_file (
              self,
              operation,
              eurl )
```

Is it allowed to create files in a certain directory?

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.3 can_create_link()

```
def shallot.Filesystem.Handler.can_create_link (
              self,
              operation,
              eurl )
```

Is it allowed to create a link in a certain directory?

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.4 can_delete_item()

```
def shallot.Filesystem.Handler.can_delete_item (
              self,
              operation,
              eurl )
```

Is it allowed to delete a certain file/directory/link/...?

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.5 can_get_filecontent()

```
def shallot.Filesystem.Handler.can_get_filecontent (
              self,
              operation,
              eurl )
```

Is it allowed to get the content of a certain file?

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.6 can_rename_item()

```
def shallot.Filesystem.Handler.can_rename_item (
              self,
              operation,
              src )
```

Is it allowed to move a certain file/directory/link/...?

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
src	The shallot.Eurl address pointing to the potential item to be moved.

1.34.2.7 configure_item()

```
def shallot.Filesystem.Handler.configure_item (
              self,
              operation,
              node )
```

Configure a newly created node (e.g.

setting another icon or changing the display name).

Override this method in custom subclasses or leave the default implementation.

Parameters

operation	The shallot.Operations.Operation operation object.
node	The shallot.Filesystem.Node filesystem node to configure.

1.34.2.8 create_directory()

```
def shallot.Filesystem.Handler.create_directory (
              self,
              operation,
              eurl )
```

Create a directory.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.9 create_file()

```
def shallot.Filesystem.Handler.create_file (
              self,
              operation,
              eurl,
              outstream,
              handlertransfer )
```

Creates a file with some content.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.
outstream	the content stream to be transferred into the new file
handlertransfer	shallot.Operations.HandlerTransfer may be used for some better integration, like cancel support and progress monitoring.

1.34.2.10 create_link()

```
def shallot.Filesystem.Handler.create_link (
              self,
              operation,
              eurl,
              tgt )
```

Create a link.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.
tgt	The link destination path (as string).

1.34.2.11 delete_item()

```
def shallot.Filesystem.Handler.delete_item (
              self,
              operation,
              eurl )
```

Delete a file/directory/link/...

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.12 get_actions()

```
def shallot.Filesystem.Handler.get_actions (
              self,
              eurls )
```

Which actions (see former example) are to be offered for certain files?

Override this method in custom subclasses.

Parameters

eurls	A list of shallot.Eurl addresses this call is referred to.

1.34.2.13 get_customattributes()

```
def shallot.Filesystem.Handler.get_customattributes (
              self,
              operation,
              eurl )
```

Returns a dict<string,string> with custom attributes for a node.

Override this method in custom subclasses or leave the default implementation. In contrast to extended attributes, those ones are not directly stored in the filesystems, but are managed in a handler specific way (e.g. file permissions).

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.14 get_extendedattribute()

```
def shallot.Filesystem.Handler.get_extendedattribute (
              self,
              operation,
              eurl,
              attribname )
```

Gets the value for a particular extended attribute.

Override this method in custom subclasses or leave the default implementation. Extended attributes are all kinds of properties of a file which aren't handled otherwise. It can contain filesystem's extended attributes, permission information and more.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.
attribname	The attribute name.

1.34.2.15 get_extendedattribute_size()

```
def shallot.Filesystem.Handler.get_extendedattribute_size (
              self,
              operation,
              eurl,
              attribname )
```

Returns the size (in byte) of the value for a particular extended attribute.

Override this method in custom subclasses or leave the default implementation. Extended attributes are all kinds of properties of a file which aren't handled otherwise. It can contain filesystem's extended attributes, permission information and more.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.
attribname	The attribute name.

1.34.2.16 get_file_content()

```
def shallot.Filesystem.Handler.get_file_content (
              self,
```

```
                operation,
                eurl )
```

Get the content of a file as a shallot.Streaming.ReadDataDevice.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.17 get_linktarget()

```
def shallot.Filesystem.Handler.get_linktarget (
                self,
                operation,
                eurl )
```

Resolve a link.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.18 get_mimetype()

```
def shallot.Filesystem.Handler.get_mimetype (
                self,
                operation,
                eurl )
```

Determine mime type of a file.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.19 get_mtime()

```
def shallot.Filesystem.Handler.get_mtime (
              self,
              operation,
              eurl )
```

Determine the mtime of a file.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.20 get_size()

```
def shallot.Filesystem.Handler.get_size (
              self,
              operation,
              eurl )
```

Determine the size of a file.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.21 get_type()

```
def shallot.Filesystem.Handler.get_type (
```

```
                self,
                operation,
                eurl )
```

Determine if a file is regular, or a dir, or a link, ...

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.22 itemlist()

```
def shallot.Filesystem.Handler.itemlist (
                self,
                operation,
                eurl,
                nodetype,
                nodelist )
```

Determine a list of subelements in a certain directory.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.
nodetype	Which kind of items are requested. See shallot.Filesystem.NodeType.
nodelist	A shallot.Filesystem.NodeList object which controls the entry list for the referred directory. Operate on this object for adding children.

1.34.2.23 list_extendedattributes()

```
def shallot.Filesystem.Handler.list_extendedattributes (
                self,
                operation,
                eurl )
```

Returns a list of extended attributes which exist for a location.

Override this method in custom subclasses or leave the default implementation. Extended attributes are all kinds of properties of a file which aren't handled otherwise. It can contain filesystem's extended attributes, permission information and more.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.

1.34.2.24 register()

```
def shallot.Filesystem.Handler.register (
              scheme,
              handler )  [static]
```

Registers a shallot.Filesystem.Handler filesystem handler implementation.

Parameters

scheme	The scheme name (first part in a shallot.Eurl, like *file*).
handler	The shallot.Filesystem.Handler filesystem handler.

1.34.2.25 remove_extendedattribute()

```
def shallot.Filesystem.Handler.remove_extendedattribute (
              self,
              operation,
              eurl,
              attribname )
```

Removes an extended attribute.

Override this method in custom subclasses or leave the default implementation. Extended attributes are all kinds of properties of a file which aren't handled otherwise. It can contain filesystem's extended attributes, permission information and more.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.
attribname	The attribute name.

1.34.2.26 rename_item()

```
def shallot.Filesystem.Handler.rename_item (
              self,
              operation,
              src,
              destpath )
```

Move a file/directory/link/...

to somewhere else (can be a simple renaming).

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
src	The shallot.Eurl address pointing to the item to be moved.
destpath	The new path for the src item.

1.34.2.27 set_customattribute()

```
def shallot.Filesystem.Handler.set_customattribute (
              self,
              operation,
              eurl,
              key,
              value )
```

Sets a custom attribute to an entry.

Override this method in custom subclasses or leave the default implementation. See also get_customattributes.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.
key	The attribute key.
value	The attribute value.

1.34.2.28 set_extendedattribute()

```
def shallot.Filesystem.Handler.set_extendedattribute (
              self,
              operation,
              eurl,
              attribname,
              value )
```

Sets the value for a particular extended attribute.

Override this method in custom subclasses or leave the default implementation. Extended attributes are all kinds of properties of a file which aren't handled otherwise. It can contain filesystem's extended attributes, permission information and more.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.
attribname	The attribute name.
value	The new attribute value (as byte string).

1.34.2.29 set_mtime()

```
def shallot.Filesystem.Handler.set_mtime (
              self,
              operation,
              eurl,
              mtime )
```

Set the mtime of a file.

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl address this call is referred to.
mtime	The datetime.datetime, which specifies the new modification time.

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

- /home/pino/projects/shallot/shallot.py

1.35 shallot.Operations.HandlerTransfer Class Reference

Used for some additional functionality in some transfer operations in shallot.Filesystem.Handler instances.

Public Member Functions

- def __init__ (self)

  Can't be constructed directly.
- def increment_transferred_bytes (self, donebytes)

  Notify that a certain amount of date (in byte) are transferred.
- def respect_cancel (self)

  Called from time to time for allowing the user to cancel the transfer.

1.35.1 Detailed Description

Used for some additional functionality in some transfer operations in shallot.Filesystem.Handler instances.

1.35.2 Member Function Documentation

1.35.2.1 increment_transferred_bytes()

```
def shallot.Operations.HandlerTransfer.increment_transferred_bytes (
              self,
              donebytes )
```

Notify that a certain amount of date (in byte) are transferred.

This is used for progress monitoring.

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

- /home/pino/projects/shallot/shallot.py

1.36 shallot.Actions.HideOnCurrentDirectoryLevelPredicate Class Reference

Shows actions only when not searching for 'current directory level' actions.

Inheritance diagram for shallot.Actions.HideOnCurrentDirectoryLevelPredicate:

Public Member Functions

- def **__init__** (self)

1.36.1 Detailed Description

Shows actions only when not searching for 'current directory level' actions.

See base class for more information.

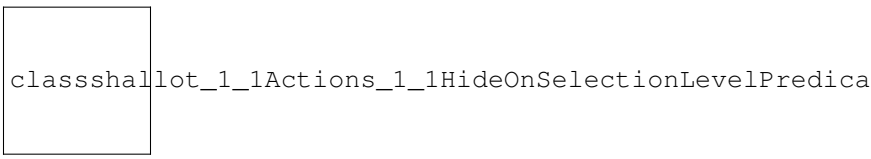
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.37 shallot.Actions.HideOnSelectionLevelPredicate Class Reference

Shows actions only when not searching for 'selection level' actions.

Inheritance diagram for shallot.Actions.HideOnSelectionLevelPredicate:



Public Member Functions

- def **__init__** (self)

1.37.1 Detailed Description

Shows actions only when not searching for 'selection level' actions.

See base class for more information.

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

- /home/pino/projects/shallot/shallot.py

1.38 shallot.IntlStringMap.IntlString Class Reference

A multi-language string.

Public Member Functions

- def __init__ (self, strings)

  Creates a multi-language string by a dictionary, considering the keys as language code and the values as the string in that language.
- def get (self)

  Returns the best variant as plain Python string, depending on the current user interface language.

1.38.1 Detailed Description

A multi-language string.

1.38.2 Constructor & Destructor Documentation

1.38.2.1 __init__()

```
def shallot.IntlStringMap.IntlString.__init__ (
              self,
              strings )
```

Creates a multi-language string by a dictionary, considering the keys as language code and the values as the string in that language.

Example: ‘shallot.IntlStringMap.IntlString({'en':'yes', 'de':'ja'})‘

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

- /home/pino/projects/shallot/shallot.py

1.39 shallot.IntlStringMap Class Reference

A multi-language string map used for localization of plugins.

Classes

- class IntlString

  A multi-language string.

Public Member Functions

- def __init__ (self, stringdicts)

  Creates a map of internationalized strings.

Public Attributes

- map

  A map of shallot.IntlStringMap.IntlString instances.

1.39.1 Detailed Description

A multi-language string map used for localization of plugins.

Internally it uses shallot.IntlStringMap.IntlString instances, but is more convenient for dealing with a larger amount of strings.

1.39.2 Constructor & Destructor Documentation

1.39.2.1 __init__()

```
def shallot.IntlStringMap.__init__ (
              self,
              stringdicts )
```

Creates a map of internationalized strings.

For each keyword-argument, this map will get a member with the key name as member name. The values are dictionaries in the same form as for shallot.IntlStringMap.IntlString. The members will be the best available language variants as plain Python strings.

Example: `Strings = shallot.IntlStringMap(Foo = {"en":"foo", "it":"Fuh"}, Bar = {"en":"bar", "it":"Barra"})`

Then `Strings.Foo` might contain the Python string "Fuh".

1.39.3 Member Data Documentation

1.39.3.1 map

```
shallot.IntlStringMap.map
```

A map of shallot.IntlStringMap.IntlString instances.

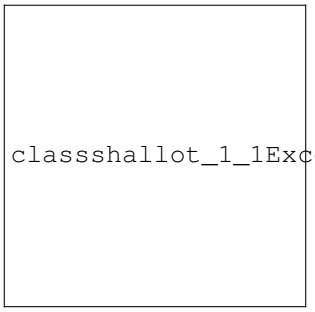
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.40 shallot.Exceptions.IOException Class Reference

Shallot exception in IO.

Inheritance diagram for shallot.Exceptions.IOException:



Public Member Functions

- def __init__ (self, details=None, message=None, isRetryable=None, autoRetryRecommendedIn=-1, detailsAreInteresting=None, _class="")

  See shallot.Exceptions.ScriptedException.__init__ for details.
- def isExceptionClass (self, classname)

  Checks if this exception is instance of a given exception class.

1.40.1 Detailed Description

Shallot exception in IO.

It allows resume and typically allows retry (special cases may override each of them). Read more about Shallot Exceptions.

1.40.2 Member Function Documentation

1.40.2.1 isExceptionClass()

```
def shallot.Exceptions.ScriptedException.isExceptionClass (
              self,
              classname )  [inherited]
```

Checks if this exception is instance of a given exception class.

Parameters

classname	A exception class name (as string).

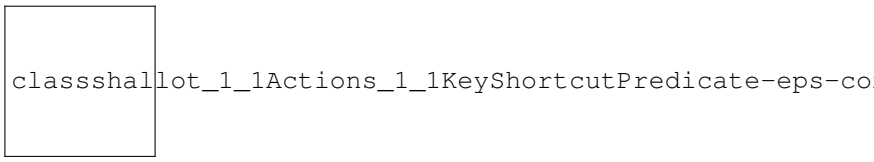
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.41 shallot.Actions.KeyShortcutPredicate Class Reference

Sets a keyboard shortcut.

Inheritance diagram for shallot.Actions.KeyShortcutPredicate:



Public Member Functions

- def **__init__** (self, shortcut, triggers_on_currentdirectory_level)

1.41.1 Detailed Description

Sets a keyboard shortcut.

See base class for more information.

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

- /home/pino/projects/shallot/shallot.py

1.42 shallot.Logging Class Reference

Methods for logging.

Static Public Member Functions

- def log_debug (s)

  Logs a message with debug severity.
- def log_info (s)

  Logs a message with info severity.
- def log_warning (s)

  Logs a message with warning severity.
- def log_error (s)

  Logs a message with message severity.

1.42.1 Detailed Description

Methods for logging.

1.42.2 Member Function Documentation

1.42.2.1 log_debug()

```
def shallot.Logging.log_debug (
              s )  [static]
```

Logs a message with debug severity.

Parameters

s	The message.

1.42.2.2 log_error()

```
def shallot.Logging.log_error (
              s )  [static]
```

Logs a message with message severity.

Parameters

s	The message.

1.42.2.3 log_info()

```
def shallot.Logging.log_info (
              s )  [static]
```

Logs a message with info severity.

Parameters

s	The message.

1.42.2.4 log_warning()

```
def shallot.Logging.log_warning (
              s )  [static]
```

Logs a message with warning severity.

Parameters

s	The message.

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

- /home/pino/projects/shallot/shallot.py

1.43 shallot.MainWindow Class Reference

A Shallot main window.

Public Member Functions

- def jump_to_eurl (self, eurl)
  Let the current view jump to another location.
- def get_current_directory_node (self)
  Gets the shallot.Filesystem.Node selected in the current view.

Static Public Member Functions

- def open_items (eurls)
  Opens a list of files, largely as if the user had doubleclicked on them.
- def current ()
  Returns the current shallot.MainWindow.

1.43.1 Detailed Description

A Shallot main window.

1.43.2 Member Function Documentation

1.43.2.1 jump_to_eurl()

```
def shallot.MainWindow.jump_to_eurl (
              self,
              eurl )
```

Let the current view jump to another location.

Parameters

eurl	The shallot.Eurl address to jump to.

1.43.2.2 open_items()

```
def shallot.MainWindow.open_items (
              eurls )  [static]
```

Opens a list of files, largely as if the user had doubleclicked on them.

Parameters

eurls	a list of shallot.Eurl.

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

- /home/pino/projects/shallot/shallot.py

1.44 shallot.Actions.ExecutionUserFeedback.MessageBoxButton Class Reference

Buttons in a message box from shallot.Actions.ExecutionUserFeedback.

Static Public Attributes

- OK = internalstuff.messageboxbutton_ok()
  OK button.
- Continue = internalstuff.messageboxbutton_continue()
  Continue button.
- Cancel = internalstuff.messageboxbutton_cancel()
  Cancel button.
- Retry = internalstuff.messageboxbutton_retry()
  Retry button.
- Yes = internalstuff.messageboxbutton_yes()
  Yes button.
- No = internalstuff.messageboxbutton_no()
  No button.

1.44.1 Detailed Description

Buttons in a message box from shallot.Actions.ExecutionUserFeedback.

1.44.2 Member Data Documentation

1.44.2.1 Cancel

```
shallot.Actions.ExecutionUserFeedback.MessageBoxButton.Cancel = internalstuff.messageboxbutton↩
_cancel()  [static]
```

Cancel button.

1.44.2.2 Continue

```
shallot.Actions.ExecutionUserFeedback.MessageBoxButton.Continue = internalstuff.messageboxbutton↩
_continue()  [static]
```

Continue button.

1.44.2.3 No

```
shallot.Actions.ExecutionUserFeedback.MessageBoxButton.No = internalstuff.messageboxbutton_↩
no()  [static]
```

No button.

1.44.2.4 OK

```
shallot.Actions.ExecutionUserFeedback.MessageBoxButton.OK = internalstuff.messageboxbutton_↩
ok()  [static]
```

OK button.

1.44.2.5 Retry

```
shallot.Actions.ExecutionUserFeedback.MessageBoxButton.Retry = internalstuff.messageboxbutton↩
_retry()  [static]
```

Retry button.

1.44.2.6 Yes

```
shallot.Actions.ExecutionUserFeedback.MessageBoxButton.Yes = internalstuff.messageboxbutton_↩
yes()  [static]
```

Yes button.

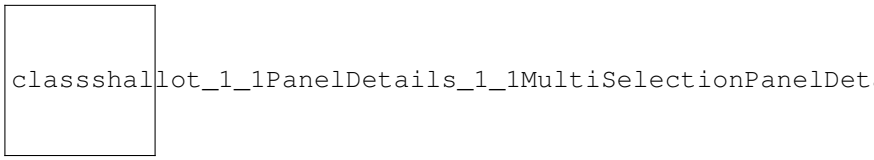
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.45 shallot.PanelDetails.MultiSelectionPanelDetail Class Reference

Abstract base class for a detail panel entry, which occurs when multiple items are selected.

Inheritance diagram for shallot.PanelDetails.MultiSelectionPanelDetail:



Public Member Functions

- def __init__ (self, positionGroup=None, positionIndex=None, valueWidthHint=4)
- def set_value (self, detail, nodes, operation)

  This method is called whenever an output must be determined for certain nodes.
- def link_triggered (self, nodes, linktarget)

  This method is called whenever the user triggers a link.

Static Public Member Functions

- def register (detail)

  Registers a shallot.PanelDetails.PanelDetail in the Shallot core.

Static Public Attributes

- INDEX_VERYINTERESTING = internalstuff.paneldetail_positionindex_veryinteresting()

  A integer value to be used as base display index for very interesting details.
- INDEX_INTERESTING = internalstuff.paneldetail_positionindex_interesting()

  A integer value to be used as base display index for interesting details.
- INDEX_EXOTIC = internalstuff.paneldetail_positionindex_exotic()

  A integer value to be used as base display index for exotic details.

1.45.1 Detailed Description

Abstract base class for a detail panel entry, which occurs when multiple items are selected.

See shallot.PanelDetails.PanelDetail.register for registering custom implementations to shallot.

1.45.2 Constructor & Destructor Documentation

1.45.2.1 __init__()

```
def shallot.PanelDetails.MultiSelectionPanelDetail.__init__ (
              self,
              positionGroup = None,
              positionIndex = None,
              valueWidthHint = 4 )
```

Parameters

positionGroup	Controls display order. See Position Indexes for details. Use one of the `INDEX_*` values from shallot.PanelDetails.PanelDetail.
positionIndex	Controls display order. See Position Indexes for details.
valueWidthHint	A width in centimeters to reserve for printing the values.

1.45.3 Member Function Documentation

1.45.3.1 link_triggered()

```
def shallot.PanelDetails.MultiSelectionPanelDetail.link_triggered (
              self,
              nodes,
              linktarget )
```

This method is called whenever the user triggers a link.

Override in custom implementations if links are used.

Parameters

nodes	The selected list of shallot.Filesystem.Node object.
linktarget	The link target string, as specified in shallot.PanelDetails.PanelDetailValueElementButton.__init__.

1.45.3.2 register()

```
def shallot.PanelDetails.PanelDetail.register (
              detail )  [static], [inherited]
```

Registers a shallot.PanelDetails.PanelDetail in the Shallot core.

Parameters

detail	The shallot.PanelDetails.PanelDetail instance.

1.45.3.3 set_value()

```
def shallot.PanelDetails.MultiSelectionPanelDetail.set_value (
              self,
              detail,
              nodes,
              operation )
```

This method is called whenever an output must be determined for certain nodes.

It must return a list of 2-tuples (one for each row). Those are each name/value pairs with the label of that row and the value (which is a list of shallot.PanelDetails.AbstractPanelDetailValueElement). For large waiting times, you should just return an empty value list for your rows, start an asynchronous execution and use shallot.PanelDetail↩ Inst.set_row in the end.

Override this method in custom subclasses.

Parameters

detail	A shallot.PanelDetailInst object for printing the details. Only used in advanced cases.
nodes	The selected list of shallot.Filesystem.Node object.
operation	The shallot.Operations.Operation operation object.

1.45.4 Member Data Documentation

1.45.4.1 INDEX_EXOTIC

```
shallot.PanelDetails.PanelDetail.INDEX_EXOTIC = internalstuff.paneldetail_positionindex_↩
exotic()  [static], [inherited]
```

A integer value to be used as base display index for exotic details.

1.45.4.2 INDEX_INTERESTING

```
shallot.PanelDetails.PanelDetail.INDEX_INTERESTING = internalstuff.paneldetail_positionindex↩
_interesting()  [static], [inherited]
```

A integer value to be used as base display index for interesting details.

1.45.4.3 INDEX_VERYINTERESTING

```
shallot.PanelDetails.PanelDetail.INDEX_VERYINTERESTING = internalstuff.paneldetail_positionindex↩
_veryinteresting()  [static], [inherited]
```

A integer value to be used as base display index for very interesting details.

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

- /home/pino/projects/shallot/shallot.py

1.46 shallot.Filesystem.Node Class Reference

A model representation of a location in the filesystem tree.

Public Member Functions

- def __init__ (self)

  Can't be constructed directly.
- def eurl (self)

  Return the shallot.Eurl entry address.
- def nodetype (self)

  Returns the shallot.Filesystem.NodeType node type.
- def add_detail (self, column)

  Adds a detail to this node.
- def set_icon (self, icon)

  Sets the node icon.
- def set_displayname (self, displayname)

  Sets the displayed name of the node.
- def ishidden (self)

  Returns if the node is hidden.
- def set_hidden (self, value)

  Sets if the node is hidden.

1.46.1 Detailed Description

A model representation of a location in the filesystem tree.

It represents stuff like a file or a directory. Those instances are used at many places for all kinds of operations.

Please note: Instances can be associated with any kind of elements in the filesystem (files, directories, links, ...). It might also point to something which does not exist at all (see parentnode). The documentation sometimes explicitly makes a difference between those kinds (files, directories, links, ...; often called 'node type'). But it often uses the term 'file' implicitly while meaning all kinds of elements; assuming that e.g. a directory is just a special kind of a file. It should be clear from the particular context which meaning applies.

1.46.2 Member Function Documentation

1.46.2.1 add_detail()

```
def shallot.Filesystem.Node.add_detail (
              self,
              column )
```

Adds a detail to this node.

Parameters

column	The shallot.DetailColumn instance to add.

1.46.2.2 ishidden()

```
def shallot.Filesystem.Node.ishidden (
              self )
```

Returns if the node is hidden.

Note: It's not difficult for the user to also show the hidden items.

1.46.2.3 set_hidden()

```
def shallot.Filesystem.Node.set_hidden (
              self,
              value )
```

Sets if the node is hidden.

See also ishidden().

Parameters

value	If the node shall be visible (boolean).

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

- /home/pino/projects/shallot/shallot.py

1.47 shallot.Filesystem.NodeList Class Reference

A list editor for a filesystem node list.

Public Member Functions

- def __init__ (self)
  Can't be constructed directly.
- def set_items (self, items)
  Sets a new list content.
- def add_item (self, item)
  Adds a new children to the list, directly showing that in the user interface, while you can proceed filling the list.
- def begin_iterative_adding (self)
  Must be called before you begin to iteratively fill the children list (e.g.
- def end_iterative_adding (self)
  Must be called after you iteratively filled the children list with shallot.Filesystem.NodeList.add_item.

1.47.1 Detailed Description

A list editor for a filesystem node list.

Thay are mainly used for specifying child nodes in shallot.Filesystem.Handler.itemlist and provide some help there. In most cases, you should just use shallot.Filesystem.NodeList.set_items.

1.47.2 Member Function Documentation

1.47.2.1 add_item()

```
def shallot.Filesystem.NodeList.add_item (
              self,
              item )
```

Adds a new children to the list, directly showing that in the user interface, while you can proceed filling the list.

Please read shallot.Filesystem.NodeList.begin_iterative_adding as well! In most cases, you don't need this function.

Parameters

item	A the new children node. This must be the basename.

1.47.2.2 begin_iterative_adding()

```
def shallot.Filesystem.NodeList.begin_iterative_adding (
              self )
```

Must be called before you begin to iteratively fill the children list (e.g.

with shallot.Filesystem.NodeList.add_item). You must also call shallot.Filesystem.NodeList.end_iterative_adding afterwards.

1.47.2.3 end_iterative_adding()

```
def shallot.Filesystem.NodeList.end_iterative_adding (
              self )
```

Must be called after you iteratively filled the children list with shallot.Filesystem.NodeList.add_item.

This automatically removes all the old nodes, which you haven't added in this session.

1.47.2.4 set_items()

```
def shallot.Filesystem.NodeList.set_items (
              self,
              items )
```

Sets a new list content.

This function is all you need in most situations. For iteratively adding nodes, which makes the intermediate results visible in the user interface, see shallot.Filesystem.NodeList.add_item.

Parameters

items	A list of strings providing the new children nodes. The list must contain the basenames of all existing children (with the node type you got as argument).

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

- /home/pino/projects/shallot/shallot.py

1.48 shallot.Filesystem.NodeType Class Reference

Enumeration of types a shallot.Filesystem.Node can have.

Static Public Attributes

- File = internalstuff.filesystemnodetype_file()

  Usual file.
- Directory = internalstuff.filesystemnodetype_directory()

  Directory.
- Link = internalstuff.filesystemnodetype_link()

  Link.
- Unknown = internalstuff.filesystemnodetype_unknown()

  Unknown.
- FIRSTTYPE = internalstuff.filesystemnodetype_firsttype()

  Points to the first type (used for generating int ranges).
- LASTPHYSICALTYPE = internalstuff.filesystemnodetype_lastphysicaltype()

  Points to the last physical type, excluding internal magic types (used for generating int ranges).
- NONE = internalstuff.filesystemnodetype_none()

  Does not exist.

1.48.1 Detailed Description

Enumeration of types a shallot.Filesystem.Node can have.

1.48.2 Member Data Documentation

1.48.2.1 Directory

```
shallot.Filesystem.NodeType.Directory = internalstuff.filesystemnodetype_directory()  [static]
```

Directory.

1.48.2.2 File

```
shallot.Filesystem.NodeType.File = internalstuff.filesystemnodetype_file()  [static]
```

Usual file.

1.48.2.3 FIRSTTYPE

```
shallot.Filesystem.NodeType.FIRSTTYPE = internalstuff.filesystemnodetype_firsttype()  [static]
```

Points to the first type (used for generating int ranges).

1.48.2.4 LASTPHYSICALTYPE

```
shallot.Filesystem.NodeType.LASTPHYSICALTYPE = internalstuff.filesystemnodetype_lastphysicaltype()
[static]
```

Points to the last physical type, excluding internal magic types (used for generating int ranges).

1.48.2.5 Link

```
shallot.Filesystem.NodeType.Link = internalstuff.filesystemnodetype_link()  [static]
```

Link.

1.48.2.6 NONE

```
shallot.Filesystem.NodeType.NONE = internalstuff.filesystemnodetype_none()  [static]
```

Does not exist.

1.48.2.7 Unknown

```
shallot.Filesystem.NodeType.Unknown = internalstuff.filesystemnodetype_unknown()  [static]
```

Unknown.

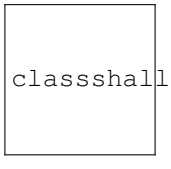
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.49 shallot.Actions.OnDirectoriesPredicate Class Reference

Shows actions only on directories.

Inheritance diagram for shallot.Actions.OnDirectoriesPredicate:



Public Member Functions

- def **__init__** (self)

1.49.1 Detailed Description

Shows actions only on directories.

See base class for more information.

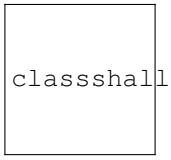
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.50 shallot.Actions.OnFilesPredicate Class Reference

Shows actions only on files.

Inheritance diagram for shallot.Actions.OnFilesPredicate:



Public Member Functions

- def **__init__** (self)

1.50.1 Detailed Description

Shows actions only on files.

See base class for more information.

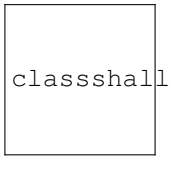
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.51 shallot.Actions.OnLinksPredicate Class Reference

Shows actions only on links.

Inheritance diagram for shallot.Actions.OnLinksPredicate:



Public Member Functions

- def **__init__** (self)

1.51.1 Detailed Description

Shows actions only on links.

See base class for more information.

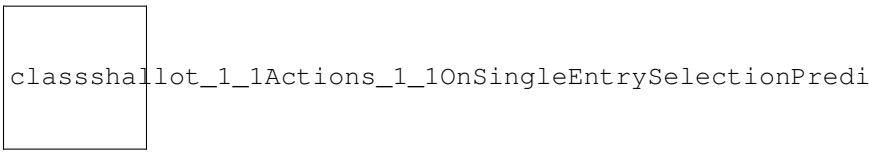
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.52 shallot.Actions.OnSingleEntrySelectionPredicate Class Reference

Shows actions only on single-entry selections.

Inheritance diagram for shallot.Actions.OnSingleEntrySelectionPredicate:



Public Member Functions

- def **__init__** (self)

1.52.1 Detailed Description

Shows actions only on single-entry selections.

See base class for more information.

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

- /home/pino/projects/shallot/shallot.py

1.53 shallot.Operations.Operation Class Reference

Transactional read and write filesystem accesses.

Public Member Functions

- def get_free_cachespace ()

  Returns the free disk space (in bytes) available for operations like fetch_container_file() or fetch_file().
- def fetch_container_file (self, eurl)

  Fetches the container file for a eurl locally and returns the local path.
- def fetch_file (self, eurl)

  Fetches a file locally and returns the local path.
- def fetch_container_file2 (self, eurl, namehint)

  Fetches the container file for a eurl locally and returns the local path.
- def fetch_file2 (self, eurl, namehint)

  Fetches a file locally and returns the local path.
- def abort (self)

  Aborts transaction dropping all pending changes.
- def commit (self)

  Commits transaction applying all pending changes.
- def enable_detailscache (self)

  Enable details caching.
- def is_detailscache_enabled (self)

  Is details caching enabled?
- def store_detail_in_cache (self, eurl, column, value)

  Stores a column detail in cache.
- def get_detail_from_cache (self, eurl, column)

  Gets a column value from cache.
- def set_custom_data (self, eurl, key, value)

  Stores custom data.
- def get_custom_data (self, eurl, key)

  Gets stored custom data.
- def filesystem (self)

  Gets the shallot.Operations.FilesystemOperation filesystem operation object.

1.53.1 Detailed Description

Transactional read and write filesystem accesses.

Read more.

1.53.2 Member Function Documentation

1.53.2.1 fetch_container_file()

```
def shallot.Operations.Operation.fetch_container_file (
              self,
              eurl )
```

Fetches the container file for a eurl locally and returns the local path.

Use this with care and only if needed. In many cases working with streams is the better way! It returns nothing for an eurl without embedded parts (like foo://[bar:///y]/x). If the outermost inner eurl represents a local file path, it is directly returned without copying. You should always regard the available disk space. See get_free_cachespace().

Parameters

eurl	The shallot.Eurl location to fetch the container for.

1.53.2.2 fetch_container_file2()

```
def shallot.Operations.Operation.fetch_container_file2 (
              self,
              eurl,
              namehint )
```

Fetches the container file for a eurl locally and returns the local path.

This is like fetch_container_file() but with more options.

Parameters

eurl	The shallot.Eurl location to fetch the container for.
namehint	A small file name prefix for temporary files, like `"tmp"`.

1.53.2.3 fetch_file()

```
def shallot.Operations.Operation.fetch_file (
              self,
              eurl )
```

Fetches a file locally and returns the local path.

Use this with care and only if needed. In many cases working with streams is the better way! If the eurl represents a local file path, it is directly returned without copying. You should always regard the available disk space. See get_free_cachespace().

Parameters

eurl	The shallot.Eurl location to fetch the container for.

1.53.2.4 fetch_file2()

```
def shallot.Operations.Operation.fetch_file2 (
              self,
              eurl,
              namehint )
```

Fetches a file locally and returns the local path.

This is like fetch_file() but with more options.

Parameters

eurl	The shallot.Eurl location to fetch the container for.
namehint	A small file name prefix for temporary files, like `"tmp"`.

1.53.2.5 get_custom_data()

```
def shallot.Operations.Operation.get_custom_data (
              self,
              eurl,
              key )
```

Gets stored custom data.

Parameters

eurl	The shallot.Eurl instance you want to get data about (or None).
key	Custom data key name.

1.53.2.6 get_detail_from_cache()

```
def shallot.Operations.Operation.get_detail_from_cache (
              self,
              eurl,
              column )
```

Gets a column value from cache.

Parameters

eurl	The shallot.Eurl to get a detail value for.
column	The shallot.DetailColumn to get a detail value for.

1.53.2.7 set_custom_data()

```
def shallot.Operations.Operation.set_custom_data (
              self,
              eurl,
```

```
                key,
                value )
```

Stores custom data.

Parameters

eurl	The shallot.Eurl instance you want to store data about (or None).
key	Custom data key name.
value	Custom data value.

1.53.2.8 store_detail_in_cache()

```
def shallot.Operations.Operation.store_detail_in_cache (
                self,
                eurl,
                column,
                value )
```

Stores a column detail in cache.

Parameters

eurl	The shallot.Eurl to store a detail value for.
column	The shallot.DetailColumn to store a detail value for.
value	The column value for this entry.

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

- /home/pino/projects/shallot/shallot.py

1.54 shallot.Operations Class Reference

Everything about operations on the filesystem.

Classes

- class FilesystemOperation

  A high-level provider of filesystem operations.
- class FilesystemOperationProgressMonitor

  Abstract base class for progress monitors, used for monitoring progress of some operations in shallot.Operations.FilesystemOperation and for conflict resolution.
- class FilesystemOperationStep

  One step in a larger filesystem transfer running inside some operations in shallot.Operations.FilesystemOperation.
- class HandlerTransfer

  Used for some additional functionality in some transfer operations in shallot.Filesystem.Handler instances.
- class Operation

  Transactional read and write filesystem accesses.

Static Public Member Functions

- def create ()

  Creates a new shallot.Operations.Operation.

1.54.1 Detailed Description

Everything about operations on the filesystem.

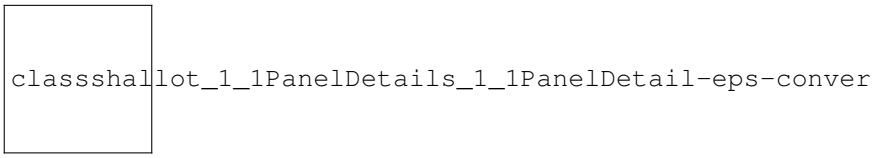
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.55 shallot.PanelDetails.PanelDetail Class Reference

Abstract base class for a detail panel entry.

Inheritance diagram for shallot.PanelDetails.PanelDetail:



Public Member Functions

- def **__init__** (self)

Static Public Member Functions

- def register (detail)

  Registers a shallot.PanelDetails.PanelDetail in the Shallot core.

Static Public Attributes

- INDEX_VERYINTERESTING = internalstuff.paneldetail_positionindex_veryinteresting()

  A integer value to be used as base display index for very interesting details.
- INDEX_INTERESTING = internalstuff.paneldetail_positionindex_interesting()

  A integer value to be used as base display index for interesting details.
- INDEX_EXOTIC = internalstuff.paneldetail_positionindex_exotic()

  A integer value to be used as base display index for exotic details.

1.55.1 Detailed Description

Abstract base class for a detail panel entry.

See the subclasses.

1.55.2 Member Function Documentation

1.55.2.1 register()

```
def shallot.PanelDetails.PanelDetail.register (
              detail )  [static]
```

Registers a shallot.PanelDetails.PanelDetail in the Shallot core.

Parameters

detail	The shallot.PanelDetails.PanelDetail instance.

1.55.3 Member Data Documentation

1.55.3.1 INDEX_EXOTIC

```
shallot.PanelDetails.PanelDetail.INDEX_EXOTIC = internalstuff.paneldetail_positionindex_↩
exotic()  [static]
```

A integer value to be used as base display index for exotic details.

1.55.3.2 INDEX_INTERESTING

```
shallot.PanelDetails.PanelDetail.INDEX_INTERESTING = internalstuff.paneldetail_positionindex↩
_interesting()  [static]
```

A integer value to be used as base display index for interesting details.

1.55.3.3 INDEX_VERYINTERESTING

```
shallot.PanelDetails.PanelDetail.INDEX_VERYINTERESTING = internalstuff.paneldetail_positionindex↩
_veryinteresting()  [static]
```

A integer value to be used as base display index for very interesting details.

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

- /home/pino/projects/shallot/shallot.py

1.56 shallot.PanelDetails.PanelDetailInst Class Reference

The storage for detail value rows, which are the actual content of a shallot.PanelDetails.PanelDetail for an actual file selection.

Public Member Functions

- def __init__ (self, _detail)
  Can't be constructed directly.
- def set_row (self, row, val)
  Sets the value for one row.

1.56.1 Detailed Description

The storage for detail value rows, which are the actual content of a shallot.PanelDetails.PanelDetail for an actual file selection.

1.56.2 Member Function Documentation

1.56.2.1 set_row()

```
def shallot.PanelDetails.PanelDetailInst.set_row (
            self,
            row,
            val )
```

Sets the value for one row.

Parameters

row	The label of that row.
val	The value (which is a list of shallot.PanelDetails.AbstractPanelDetailValueElement).

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

- /home/pino/projects/shallot/shallot.py

1.57 shallot.PanelDetails Class Reference

Everything about panel details.

Classes

- class AbstractPanelDetailValueElement

  Abstract base class for an element of a panel detail row's value.
- class MultiSelectionPanelDetail

  Abstract base class for a detail panel entry, which occurs when multiple items are selected.
- class PanelDetail

  Abstract base class for a detail panel entry.
- class PanelDetailInst

  The storage for detail value rows, which are the actual content of a shallot.PanelDetails.PanelDetail for an actual file selection.
- class PanelDetailValueElementButton

  A link button (for executing some code on user behalf) as element of a panel detail row's value.
- class PanelDetailValueElementIcon

  An icon as element of a panel detail row's value.
- class PanelDetailValueElementString

  A piece of text as element of a panel detail row's value.
- class PanelDetailValueElementWaiting

  A 'waiting' placeholder as element of a panel detail row's value.
- class SingleSelectionPanelDetail

  Abstract base class for a detail panel entry, which occurs when one item is selected.

1.57.1 Detailed Description

Everything about panel details.

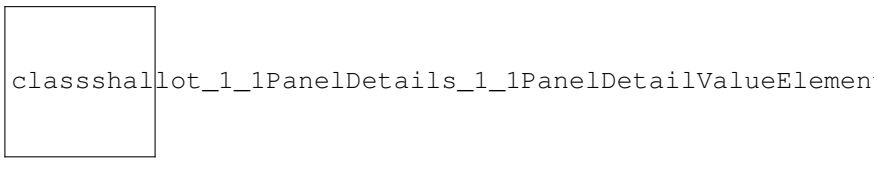
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.58 shallot.PanelDetails.PanelDetailValueElementButton Class Reference

A link button (for executing some code on user behalf) as element of a panel detail row's value.

Inheritance diagram for shallot.PanelDetails.PanelDetailValueElementButton:



Public Member Functions

- def __init__ (self, text, target, autohide=True)

Public Attributes

- **type**
- **value**

1.58.1 Detailed Description

A link button (for executing some code on user behalf) as element of a panel detail row's value.

Use shallot.PanelDetails.SingleSelectionPanelDetail.link_triggered or shallot.PanelDetails.MultiSelectionPanel↩ Detail.link_triggered for handling user actions.

1.58.2 Constructor & Destructor Documentation

1.58.2.1 __init__()

```
def shallot.PanelDetails.PanelDetailValueElementButton.__init__ (
              self,
              text,
              target,
              autohide = True )
```

Parameters

text	The text.
target	The link target name (will be passed to the handler function).
autohide	If the button shall only be visible when the mouse cursor is in the details panel.

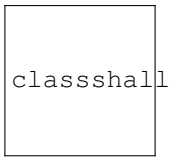
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.59 shallot.PanelDetails.PanelDetailValueElementIcon Class Reference

An icon as element of a panel detail row's value.

Inheritance diagram for shallot.PanelDetails.PanelDetailValueElementIcon:



Public Member Functions

- def __init__ (self, icon)

Public Attributes

- **type**
- **value**

1.59.1 Detailed Description

An icon as element of a panel detail row's value.

1.59.2 Constructor & Destructor Documentation

1.59.2.1 __init__()

```
def shallot.PanelDetails.PanelDetailValueElementIcon.__init__ (
              self,
              icon )
```

Parameters

icon	The icon name.

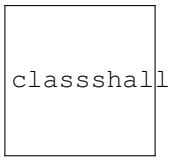
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.60 shallot.PanelDetails.PanelDetailValueElementString Class Reference

A piece of text as element of a panel detail row's value.

Inheritance diagram for shallot.PanelDetails.PanelDetailValueElementString:



Public Member Functions

- def __init__ (self, text)

Public Attributes

- **type**
- **value**

1.60.1 Detailed Description

A piece of text as element of a panel detail row's value.

1.60.2 Constructor & Destructor Documentation

1.60.2.1 __init__()

```
def shallot.PanelDetails.PanelDetailValueElementString.__init__ (
              self,
              text )
```

Parameters

text	The text.

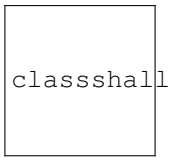
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.61 shallot.PanelDetails.PanelDetailValueElementWaiting Class Reference

A 'waiting' placeholder as element of a panel detail row's value.

Inheritance diagram for shallot.PanelDetails.PanelDetailValueElementWaiting:



Public Member Functions

- def **__init__** (self)

Public Attributes

- **type**
- **value**

1.61.1 Detailed Description

A 'waiting' placeholder as element of a panel detail row's value.

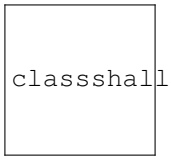
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.62 shallot.Actions.PositionIndexPredicate Class Reference

Sets a positioning information index.

Inheritance diagram for shallot.Actions.PositionIndexPredicate:



Public Member Functions

- def **__init__** (self, index)

1.62.1 Detailed Description

Sets a positioning information index.

See base class for more information.

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

- /home/pino/projects/shallot/shallot.py

1.63 shallot.Actions.Predicate Class Reference

Controls when and how an action is created.

Inheritance diagram for shallot.Actions.Predicate:

1.63.1 Detailed Description

Controls when and how an action is created.

Used in shallot.Actions.register. See base class for more information.

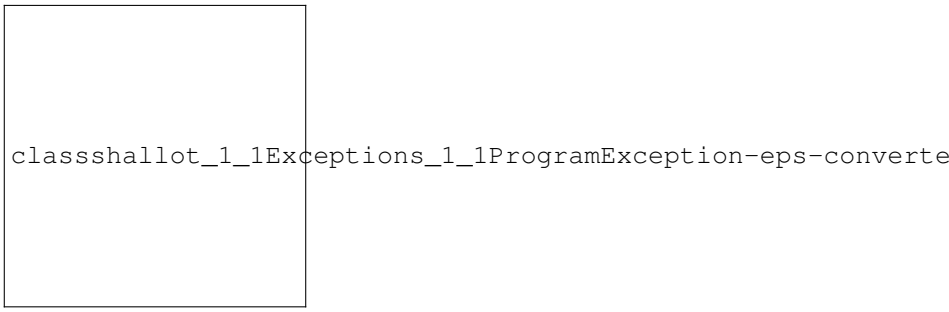
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.64 shallot.Exceptions.ProgramException Class Reference

Shallot exception for program errors (typically logical stuff) in the program or plugin.

Inheritance diagram for shallot.Exceptions.ProgramException:



Public Member Functions

- def __init__ (self, details=None, message=None, isResumeable=True, detailsAreInteresting=None, _↩ class="")

  See shallot.Exceptions.ScriptedException.__init__ for details.
- def isExceptionClass (self, classname)

  Checks if this exception is instance of a given exception class.

1.64.1 Detailed Description

Shallot exception for program errors (typically logical stuff) in the program or plugin.

It typically allows resume but no retry (special cases may override each of them). Read more about Shallot Exceptions.

1.64.2 Member Function Documentation

1.64.2.1 isExceptionClass()

```
def shallot.Exceptions.ScriptedException.isExceptionClass (
              self,
              classname )  [inherited]
```

Checks if this exception is instance of a given exception class.

Parameters

classname	A exception class name (as string).

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

- /home/pino/projects/shallot/shallot.py

1.65 shallot.FilePropertyDialog.Tab.PropertyButtonConfig Class Reference

Definitions for buttons in a shallot.FilePropertyDialog.Tab.PropertyConfig.

Public Member Functions

- def __init__ (self, label, fct)

Public Attributes

- **label**
- **fct**
- **id**

1.65.1 Detailed Description

Definitions for buttons in a shallot.FilePropertyDialog.Tab.PropertyConfig.

1.65.2 Constructor & Destructor Documentation

1.65.2.1 __init__()

```
def shallot.FilePropertyDialog.Tab.PropertyButtonConfig.__init__ (
              self,
              label,
              fct )
```

Parameters

label	The button text.
fct	The function to execute when the user clicks on the button.

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

- /home/pino/projects/shallot/shallot.py

1.66 shallot.FilePropertyDialog.Tab.PropertyConfig Class Reference

Definitions for properties, which present a piece of information in a shallot.FilePropertyDialog.Tab.

Public Member Functions

- def __init__ (self, title, propertytype, buttons=[ ])

Public Attributes

- **title**
- **propertytype**
- **buttons**

1.66.1 Detailed Description

Definitions for properties, which present a piece of information in a shallot.FilePropertyDialog.Tab.

1.66.2 Constructor & Destructor Documentation

1.66.2.1 __init__()

```
def shallot.FilePropertyDialog.Tab.PropertyConfig.__init__ (
              self,
              title,
              propertytype,
              buttons = [] )
```

Parameters

title	The property title/name.
propertytype	The value type. This controls what you have to return in shallot.FilePropertyDialog.Tab.update_widget and how it is displayed. See the PROPERTYTYPE_∗ constants.
buttons	A list of shallot.FilePropertyDialog.Tab.PropertyButtonConfig specifying which buttons to show for offering user interaction.

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

- /home/pino/projects/shallot/shallot.py

1.67 shallot.FilePropertyDialog.Tab.PropertyType Class Reference

A type of a single property.

Static Public Attributes

- String = internalstuff.filepropertydialogtab_propertytype_string()
  A single string.
- StringMap = internalstuff.filepropertydialogtab_propertytype_stringmap()
  A table of key/value string pairs.
- IconTextBanner = internalstuff.filepropertydialogtab_propertytype_icontextbanner()
  A mixed banner with texts and icons in a horizontal alignment.

1.67.1 Detailed Description

A type of a single property.

Used in shallot.FilePropertyDialog.Tab.PropertyConfig.

1.67.2 Member Data Documentation

1.67.2.1 IconTextBanner

```
shallot.FilePropertyDialog.Tab.PropertyType.IconTextBanner = internalstuff.filepropertydialogtab↩
_propertytype_icontextbanner()  [static]
```

A mixed banner with texts and icons in a horizontal alignment.

If chosen, return a shallot.FilePropertyDialog.TabPropertyIconTextBanner in update_widget.

1.67.2.2 String

```
shallot.FilePropertyDialog.Tab.PropertyType.String = internalstuff.filepropertydialogtab_↩
propertytype_string()  [static]
```

A single string.

If chosen, return a string in update_widget.

1.67.2.3 StringMap

```
shallot.FilePropertyDialog.Tab.PropertyType.StringMap = internalstuff.filepropertydialogtab_↩
propertytype_stringmap()  [static]
```

A table of key/value string pairs.

If chosen, return a list of (string,string) tuples in update_widget.

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

- /home/pino/projects/shallot/shallot.py

1.68 shallot.Streaming.ReadDataDevice Class Reference

A scripted source of binary content.

Inheritance diagram for shallot.Streaming.ReadDataDevice:

Public Member Functions

- def **__init__** (self)
- def getdata (self)

  Returns a Python byte array containing the next available chunk of content data.
- def read (self)

  Reads a chuck of data as Python byte array.
- def readall (self)

  Reads the complete data as Python byte array.

1.68.1 Detailed Description

A scripted source of binary content.

Override this class for providing own binary content. You might often find it more convenient to use one of the non-abstract subclasses.

1.68.2 Member Function Documentation

1.68.2.1 getdata()

```
def shallot.Streaming.ReadDataDevice.getdata (
              self )
```

Returns a Python byte array containing the next available chunk of content data.

This may either be the complete content, or just the next part in an arbitrary size which is convenient in your situation. It is allowed to return empty arrays whenever there is temporarily no data available. Return `None` when the end of stream is reached.

Override this method in custom subclasses.

1.68.2.2 read()

```
def shallot.Streaming.ReadDataDevice.read (
              self )
```

Reads a chuck of data as Python byte array.

An empty array signals the stream ended.

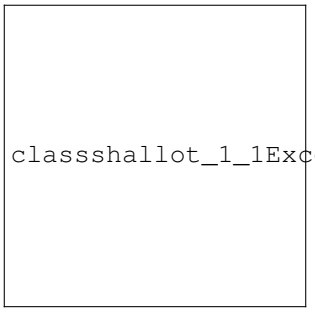
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.69 shallot.Exceptions.RuntimeException Class Reference

Shallot exception for failed operation due to (often external) runtime effects.

Inheritance diagram for shallot.Exceptions.RuntimeException:



Public Member Functions

- def __init__ (self, details=None, message=None, isRetryable=None, autoRetryRecommendedIn=-1, detailsAreInteresting=None, _class="")

  See shallot.Exceptions.ScriptedException.__init__ for details.
- def isExceptionClass (self, classname)

  Checks if this exception is instance of a given exception class.

1.69.1 Detailed Description

Shallot exception for failed operation due to (often external) runtime effects.

It allows resume and optionally allows retry (special cases may override each of them). Read more about Shallot Exceptions.

1.69.2 Member Function Documentation

1.69.2.1 isExceptionClass()

```
def shallot.Exceptions.ScriptedException.isExceptionClass (
              self,
              classname )  [inherited]
```

Checks if this exception is instance of a given exception class.

Parameters

classname	A exception class name (as string).

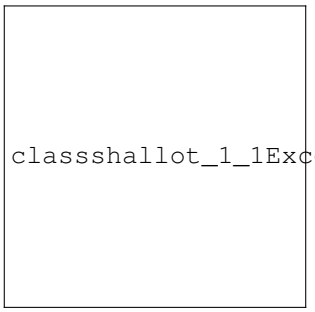
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.70 shallot.Exceptions.ScriptedException Class Reference

The Shallot exception class.

Inheritance diagram for shallot.Exceptions.ScriptedException:



Public Member Functions

- def __init__ (self, details=None, message=None, isRuntime=None, isResumeable=None, isRetryable=None, autoRetryRecommendedIn=-1, detailsAreInteresting=None, _class="")
- def isExceptionClass (self, classname)

  Checks if this exception is instance of a given exception class.

1.70.1 Detailed Description

The Shallot exception class.

1.70.2 Constructor & Destructor Documentation

1.70.2.1 __init__()

```
def shallot.Exceptions.ScriptedException.__init__ (
              self,
              details = None,
              message = None,
              isRuntime = None,
              isResumeable = None,
              isRetryable = None,
              autoRetryRecommendedIn = -1,
              detailsAreInteresting = None,
              _class = "" )
```

Parameters

details	Detail text.
message	Message text.
isRuntime	Is a runtime error (instead of a program logic error).
isResumeable	Is resumeable.
isRetryable	Is retryable.
detailsAreInteresting	If the details contain information which is directly interesting for (and consumable by) the end user.
autoRetry↩ RecommendedIn	Recommended retry interval in milliseconds.
_class	Only used internally.

1.70.3 Member Function Documentation

1.70.3.1 isExceptionClass()

```
def shallot.Exceptions.ScriptedException.isExceptionClass (
              self,
              classname )
```

Checks if this exception is instance of a given exception class.

Parameters

classname	A exception class name (as string).

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

- /home/pino/projects/shallot/shallot.py

1.71 shallot.FileSearch.SearchCriterion Class Reference

Abstract base class for a search criterion.

Inheritance diagram for shallot.FileSearch.SearchCriterion:

Public Member Functions

- def __init__ (self, factory)

  Each implementation must offer a constructor with this signature.
- def configure (self, info)

  Asks some questions to the user (with `info` parameter) in order to determine a configuration and returns it as a list of strings.
- def searchspec (self)

  Returns the current search configuration as string list.
- def match (self, operation, eurl)

  Determines if a file matches the configured search criteria (returns bool).

1.71.1 Detailed Description

Abstract base class for a search criterion.

It implements custom file search filters. Implement this class as well as shallot.FileSearch.SearchCriterionFactory and register this pair with shallot.FileSearch.SearchCriterionFactory.register.

1.71.2 Constructor & Destructor Documentation

1.71.2.1 __init__()

```
def shallot.FileSearch.SearchCriterion.__init__ (
              self,
              factory )
```

Each implementation must offer a constructor with this signature.

It must forward the `factory` parameter to this constructor.

1.71.3 Member Function Documentation

1.71.3.1 configure()

```
def shallot.FileSearch.SearchCriterion.configure (
              self,
              info )
```

Asks some questions to the user (with `info` parameter) in order to determine a configuration and returns it as a list of strings.

Some other methods must be able to interpret this list in order to get back this configuration.

Override this method in custom subclasses.

Parameters

info	A shallot.Actions.ExecutionInfo execution info object.

1.71.3.2 match()

```
def shallot.FileSearch.SearchCriterion.match (
              self,
              operation,
              eurl )
```

Determines if a file matches the configured search criteria (returns bool).

Override this method in custom subclasses. Use searchspec for getting the current configuration.

Parameters

operation	The shallot.Operations.Operation operation object.
eurl	The shallot.Eurl on which the filter logic must act.

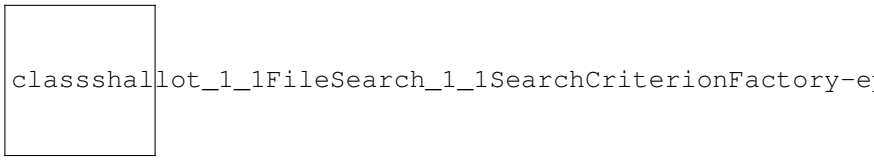
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.72 shallot.FileSearch.SearchCriterionFactory Class Reference

A factory for a shallot.FileSearch.SearchCriterion class.

Inheritance diagram for shallot.FileSearch.SearchCriterionFactory:



Public Member Functions

- def __init__ (self, key, ctype, description)

  Call this constructor from subclasses.
- def get_searchspec_description (self, val)

  Returns the description of a search configuration (for displaying in the user interface).
- def is_visible_for (self, operation, node)

  Determines if this search criterion should be offered for a node.

Static Public Member Functions

- def register (criterionfactory, positionGroup=None, positionIndex=None)

  Registers a custom shallot.FileSearch.SearchCriterionFactory instance for offering custom file searching filters.

Static Public Attributes

- INDEX_CORE = internalstuff.searchcriterionfactory_index_core()

  Base index value for core (i.e.
- INDEX_NORMAL = internalstuff.searchcriterionfactory_index_normal()

  Base index value for search criteria with normal importance.
- INDEX_EXOTIC = internalstuff.searchcriterionfactory_index_exotic()

  Base index value for exotic search criteria.

1.72.1 Detailed Description

A factory for a shallot.FileSearch.SearchCriterion class.

1.72.2 Constructor & Destructor Documentation

1.72.2.1 __init__()

```
def shallot.FileSearch.SearchCriterionFactory.__init__ (
              self,
              key,
              ctype,
              description )
```

Call this constructor from subclasses.

Parameters

key	A short string used as key for this criterion.
ctype	The type of your shallot.FileSearch.SearchCriterion implementation to create.
description	The short description (as shown in menus).

1.72.3 Member Function Documentation

1.72.3.1 get_searchspec_description()

```
def shallot.FileSearch.SearchCriterionFactory.get_searchspec_description (
              self,
```

```
                val )
```

Returns the description of a search configuration (for displaying in the user interface).

Override this method in custom subclasses or leave the default implementation.

Parameters

val	The search configuration as string list.

1.72.3.2 is_visible_for()

```
def shallot.FileSearch.SearchCriterionFactory.is_visible_for (
                self,
                operation,
                node )
```

Determines if this search criterion should be offered for a node.

Override this method in custom subclasses or leave the default implementation.

Parameters

operation	The shallot.Operations.Operation operation object.
node	The shallot.Filesystem.Node node for which the visibility must be determined.

1.72.3.3 register()

```
def shallot.FileSearch.SearchCriterionFactory.register (
                criterionfactory,
                positionGroup = None,
                positionIndex = None )  [static]
```

Registers a custom shallot.FileSearch.SearchCriterionFactory instance for offering custom file searching filters.

Parameters

criterionfactory	An instance of shallot.FileSearch.SearchCriterionFactory.
positionGroup	Controls display order. See Position Indexes for details. Use one of the `INDEX_*` values from shallot.FileSearch.SearchCriterionFactory.
positionIndex	Controls display order. See Position Indexes for details.

1.72.4 Member Data Documentation

1.72.4.1 INDEX_CORE

```
shallot.FileSearch.SearchCriterionFactory.INDEX_CORE = internalstuff.searchcriterionfactory_↩
index_core()  [static]
```

Base index value for core (i.e.

very interesting) search criteria.

1.72.4.2 INDEX_EXOTIC

```
shallot.FileSearch.SearchCriterionFactory.INDEX_EXOTIC = internalstuff.searchcriterionfactory↩
_index_exotic()  [static]
```

Base index value for exotic search criteria.

1.72.4.3 INDEX_NORMAL

```
shallot.FileSearch.SearchCriterionFactory.INDEX_NORMAL = internalstuff.searchcriterionfactory↩
_index_normal()  [static]
```

Base index value for search criteria with normal importance.

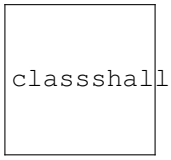
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.73 shallot.Setting Class Reference

Abstract base class for a scripted Shallot setting (those who have to be stored manually).

Inheritance diagram for shallot.Setting:



Classes

- class GroupInfo

  Enumeration of groups to which a setting can belong.

Public Member Functions

- def __init__ (self, name, description, group, isAdvancedSetting=False, isGlobal=False, isPerFileview=False)
- def set_value (self, value, viewindex=None)

  Called from Shallot core when the value was set.
- def get_value (self, viewindex)

  Get the currently set value.
- def value_description (self, val)

  Gets a human readable description text for a value.

Static Public Member Functions

- def register (setting)

  Registers a shallot.Setting instance in Shallot.

1.73.1 Detailed Description

Abstract base class for a scripted Shallot setting (those who have to be stored manually).

See Shallot documentation for details.

1.73.2 Constructor & Destructor Documentation

1.73.2.1 __init__()

```
def shallot.Setting.__init__ (
              self,
              name,
              description,
              group,
              isAdvancedSetting = False,
              isGlobal = False,
              isPerFileview = False )
```

Parameters

name	Internal name (must be unique).
description	Description text.
group	Group. One of shallot.Setting.GroupInfo.
isAdvancedSetting	Is this an advanced setting?
isGlobal	Does this setting regard global aspects (instead of per-directory aspects)?
isPerFileview	Does this setting regard per-fileview aspects (instead of per-mainwindow)?

1.73.3 Member Function Documentation

1.73.3.1 get_value()

```
def shallot.Setting.get_value (
              self,
              viewindex )
```

Get the currently set value.

Override this method in custom subclasses.

Parameters

viewindex	View index. Will be `Nothing` for per-mainwindow settings. Otherwise a fileview index.

1.73.3.2 register()

```
def shallot.Setting.register (
              setting )  [static]
```

Registers a shallot.Setting instance in Shallot.

Parameters

setting	The shallot.Setting setting object.

1.73.3.3 set_value()

```
def shallot.Setting.set_value (
              self,
              value,
              viewindex = None )
```

Called from Shallot core when the value was set.

Override this method in custom subclasses.

Parameters

value	The value.
viewindex	View index. Will be `Nothing` for per-mainwindow settings. Otherwise a fileview index.

1.73.3.4 value_description()

```
def shallot.Setting.value_description (
              self,
              val )
```

Gets a human readable description text for a value.

Override this method in custom subclasses.

Parameters

val	The value.

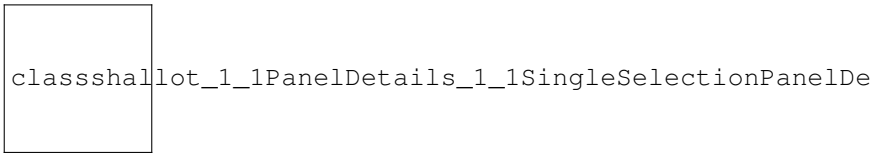
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.74 shallot.PanelDetails.SingleSelectionPanelDetail Class Reference

Abstract base class for a detail panel entry, which occurs when one item is selected.

Inheritance diagram for shallot.PanelDetails.SingleSelectionPanelDetail:



Public Member Functions

- def __init__ (self, positionGroup=None, positionIndex=None, valueWidthHint=4)
- def set_value (self, detail, node, operation)

  This method is called whenever an output must be determined for a certain node.
- def link_triggered (self, node, linktarget)

  This method is called whenever the user triggers a link.

Static Public Member Functions

- def register (detail)

  Registers a shallot.PanelDetails.PanelDetail in the Shallot core.

Static Public Attributes

- INDEX_VERYINTERESTING = internalstuff.paneldetail_positionindex_veryinteresting()

  A integer value to be used as base display index for very interesting details.
- INDEX_INTERESTING = internalstuff.paneldetail_positionindex_interesting()

  A integer value to be used as base display index for interesting details.
- INDEX_EXOTIC = internalstuff.paneldetail_positionindex_exotic()

  A integer value to be used as base display index for exotic details.

1.74.1 Detailed Description

Abstract base class for a detail panel entry, which occurs when one item is selected.

See shallot.PanelDetails.PanelDetail.register for registering custom implementations to shallot.

1.74.2 Constructor & Destructor Documentation

1.74.2.1 __init__()

```
def shallot.PanelDetails.SingleSelectionPanelDetail.__init__ (
              self,
              positionGroup = None,
              positionIndex = None,
              valueWidthHint = 4 )
```

Parameters

positionGroup	Controls display order. See Position Indexes for details. Use one of the `INDEX_*` values from shallot.PanelDetails.PanelDetail.
positionIndex	Controls display order. See Position Indexes for details.
valueWidthHint	A width in centimeters to reserve for printing the values.

1.74.3 Member Function Documentation

1.74.3.1 link_triggered()

```
def shallot.PanelDetails.SingleSelectionPanelDetail.link_triggered (
              self,
              node,
              linktarget )
```

This method is called whenever the user triggers a link.

Override in custom implementations if links are used.

Parameters

node	The selected shallot.Filesystem.Node.
linktarget	The link target string, as specified in shallot.PanelDetails.PanelDetailValueElementButton.__init__.

1.74.3.2 register()

```
def shallot.PanelDetails.PanelDetail.register (
              detail )  [static], [inherited]
```

Registers a shallot.PanelDetails.PanelDetail in the Shallot core.

Parameters

detail	The shallot.PanelDetails.PanelDetail instance.

1.74.3.3 set_value()

```
def shallot.PanelDetails.SingleSelectionPanelDetail.set_value (
              self,
              detail,
              node,
              operation )
```

This method is called whenever an output must be determined for a certain node.

It must return a list of 2-tuples (one for each row). Those are each name/value pairs with the label of that row and the value (which is a list of shallot.PanelDetails.PanelDetails.AbstractPanelDetailValueElement). For large waiting times, you should just return a value list only containing a shallot.PanelDetails.PanelDetailValueElementWaiting for your rows, start an asynchronous execution and use shallot.PanelDetailInst.set_row in the end.

Override this method in custom subclasses.

Parameters

detail	A shallot.PanelDetailInst object for printing the details. Only used in advanced cases.
node	The selected shallot.Filesystem.Node.
operation	The shallot.Operations.Operation operation object.

1.74.4 Member Data Documentation

1.74.4.1 INDEX_EXOTIC

```
shallot.PanelDetails.PanelDetail.INDEX_EXOTIC = internalstuff.paneldetail_positionindex_↩
exotic()  [static], [inherited]
```

A integer value to be used as base display index for exotic details.

1.74.4.2 INDEX_INTERESTING

```
shallot.PanelDetails.PanelDetail.INDEX_INTERESTING = internalstuff.paneldetail_positionindex↩
_interesting()  [static], [inherited]
```

A integer value to be used as base display index for interesting details.

1.74.4.3 INDEX_VERYINTERESTING

```
shallot.PanelDetails.PanelDetail.INDEX_VERYINTERESTING = internalstuff.paneldetail_positionindex↩
_veryinteresting()  [static], [inherited]
```

A integer value to be used as base display index for very interesting details.

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

- /home/pino/projects/shallot/shallot.py

1.75 shallot.Streaming Class Reference

Streams of binary data.

Classes

- class BlobReadDataDevice

  A binary source backed by static binary content in a byte array.
- class ReadDataDevice

  A scripted source of binary content.
- class StreamReadDataDevice

  A binary source backed by a file object.
- class ThreadedReadDataDevice

  A binary source dynamically created piecewise in a separate thread.

1.75.1 Detailed Description

Streams of binary data.

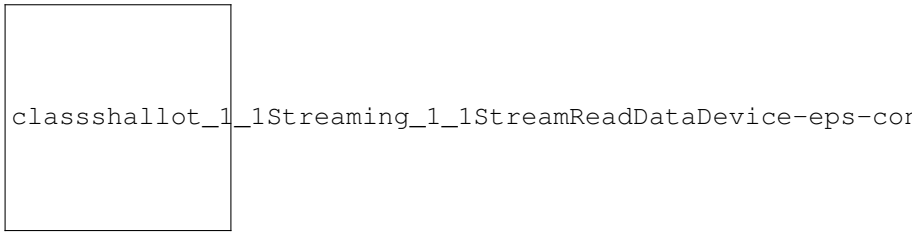
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.76 shallot.Streaming.StreamReadDataDevice Class Reference

A binary source backed by a file object.

Inheritance diagram for shallot.Streaming.StreamReadDataDevice:



Public Member Functions

- def __init__ (self, stream)
- def getdata (self)

  See shallot.Streaming.ReadDataDevice.
- def read (self)

  Reads a chuck of data as Python byte array.
- def readall (self)

  Reads the complete data as Python byte array.

Public Attributes

- **stream**

1.76.1 Detailed Description

A binary source backed by a file object.

The file object must be opened in blocking mode (which is the typical one).

1.76.2 Constructor & Destructor Documentation

1.76.2.1 __init__()

```
def shallot.Streaming.StreamReadDataDevice.__init__ (
              self,
              stream )
```

Parameters

stream	The file object containing the content.

1.76.3 Member Function Documentation

1.76.3.1 read()

```
def shallot.Streaming.ReadDataDevice.read (
              self )  [inherited]
```

Reads a chuck of data as Python byte array.

An empty array signals the stream ended.

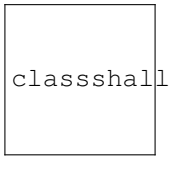
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.77 shallot.Actions.SubmenuAction Class Reference

Abstract base class for a submenu action, which can be made visible in menus or the toolbar.

Inheritance diagram for shallot.Actions.SubmenuAction:



Public Member Functions

- def __init__ (self, text, enabled=True, icon="", defaultActionPrecedence=0)
- def set_subitems (self, subitems)

  Sets the subitems.
- def initialize (self)

  Initialize the action.
- def set_enabled (self, value)

  Sets if the item is enabled.
- def set_visible (self, value)

  Sets the visibility of this item.
- def enabled (self)

  Checks if this action is enabled.
- def visible (self)

  Checks the visibility of this item (non-recursively).

1.77.1 Detailed Description

Abstract base class for a submenu action, which can be made visible in menus or the toolbar.

See shallot.Actions.AbstractAction for more.

1.77.2 Constructor & Destructor Documentation

1.77.2.1 __init__()

```
def shallot.Actions.SubmenuAction.__init__ (
              self,
              text,
              enabled = True,
              icon = "",
              defaultActionPrecedence = 0 )
```

Parameters

text	A label text.
enabled	If this action is enabled.
icon	Name of an icon.
defaultActionPrecedence	Precedence value for being a default action. This int value must be higher than all others for becoming the default action. See shallot.Actions.DefaultPrecedenceValues.

1.77.3 Member Function Documentation

1.77.3.1 initialize()

```
def shallot.Actions.AbstractAction.initialize (
              self )  [inherited]
```

Initialize the action.

This should make the time-consuming parts, e.g. for determining a label or enabled state.

Override this method in custom subclasses or leave the default implementation.

1.77.3.2 set_enabled()

```
def shallot.Actions.AbstractAction.set_enabled (
              self,
              value )  [inherited]
```

Sets if the item is enabled.

Parameters

value	The new value.

1.77.3.3 set_subitems()

```
def shallot.Actions.SubmenuAction.set_subitems (
              self,
              subitems )
```

Sets the subitems.

Parameters

subitems	A list of shallot.Actions.AbstractAction instances.

1.77.3.4 set_visible()

```
def shallot.Actions.AbstractAction.set_visible (
              self,
              value )  [inherited]
```

Sets the visibility of this item.

Parameters

value	The new value.

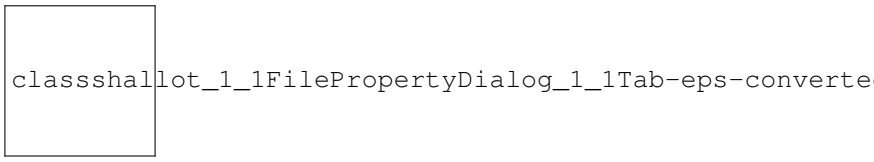
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.78 shallot.FilePropertyDialog.Tab Class Reference

Abstract base class for a tab in the Properties dialog.

Inheritance diagram for shallot.FilePropertyDialog.Tab:



Classes

- class PropertyButtonConfig

  Definitions for buttons in a shallot.FilePropertyDialog.Tab.PropertyConfig.
- class PropertyConfig

  Definitions for properties, which present a piece of information in a shallot.FilePropertyDialog.Tab.
- class PropertyType

  A type of a single property.

Public Member Functions

- def __init__ (self, title, properties)
- def refresh (self)

  Reloads the content in the dialog (typically called after the external situation has changed so the dialog shows outdated information).
- def selected_index_for_property (self, index)

  For a shallot.FilePropertyDialog.Tab.PropertyType.StringMap property, this returns the index of the row selected by the user in the dialog (or -1 for no selection).
- def nodes (self)

  Returns a list of shallot.Filesystem.Node containing the nodes to show (typically one).
- def update_widget (self, i, operation)

  Provides the actual content.

Static Public Member Functions

- def register (tabclass, positionGroup=None, positionIndex=None)

  Registers a shallot.FilePropertyDialog.Tab implementation for making its content available in the Properties dialog.

Public Attributes

- **properties**
- **btnid2btn**

Static Public Attributes

- INDEX_CORE = internalstuff.filepropertydialogtab_index_core()

  An integer value to be used as base value for specifying order indexes of core (i.e.
- INDEX_VERYIMPORTANT = internalstuff.filepropertydialogtab_index_veryimportant()

  An integer value to be used as base value for specifying order indexes of very important properties.
- INDEX_IMPORTANT = internalstuff.filepropertydialogtab_index_important()

  An integer value to be used as base value for specifying order indexes of important properties.
- INDEX_NORMAL = internalstuff.filepropertydialogtab_index_normal()

  An integer value to be used as base value for specifying order indexes of properties with normal importance.
- INDEX_EXOTIC = internalstuff.filepropertydialogtab_index_exotic()

  An integer value to be used as base value for specifying order indexes of exotic (i.e.

1.78.1 Detailed Description

Abstract base class for a tab in the Properties dialog.

It provides content and can also offer user interactions. See shallot.FilePropertyDialog.Tab.register for registering custom implementations to shallot.

1.78.2 Constructor & Destructor Documentation

1.78.2.1 __init__()

```
def shallot.FilePropertyDialog.Tab.__init__ (
              self,
              title,
              properties )
```

Parameters

title	The title text for this tab.
properties	A list of shallot.FilePropertyDialog.Tab.PropertyConfig instances. Each instances specifies one piece of information you want to provide.

1.78.3 Member Function Documentation

1.78.3.1 register()

```
def shallot.FilePropertyDialog.Tab.register (
              tabclass,
              positionGroup = None,
              positionIndex = None )  [static]
```

Registers a shallot.FilePropertyDialog.Tab implementation for making its content available in the Properties dialog.

Parameters

positionGroup	Controls display order. See Position Indexes for details. Use one of the `INDEX_*` values from shallot.FilePropertyDialog.Tab.
positionIndex	Controls display order. See Position Indexes for details.
tabclass	A subclass of shallot.FilePropertyDialog.Tab.

1.78.3.2 selected_index_for_property()

```
def shallot.FilePropertyDialog.Tab.selected_index_for_property (
              self,
              index )
```

For a shallot.FilePropertyDialog.Tab.PropertyType.StringMap property, this returns the index of the row selected by the user in the dialog (or -1 for no selection).

It is not allowed to call it for other properties.

1.78.3.3 update_widget()

```
def shallot.FilePropertyDialog.Tab.update_widget (
              self,
              i,
              operation )
```

Provides the actual content.

Returns a value depending on the type choice in shallot.FilePropertyDialog.Tab.PropertyConfig.

Override this method in custom subclasses.

Parameters

i	The index (in the same order as the `properties` constructor parameter).
operation	A shallot.Operations.Operation instance.

1.78.4 Member Data Documentation

1.78.4.1 INDEX_CORE

```
shallot.FilePropertyDialog.Tab.INDEX_CORE = internalstuff.filepropertydialogtab_index_core()
[static]
```

An integer value to be used as base value for specifying order indexes of core (i.e.

maximum important) properties.

1.78.4.2 INDEX_EXOTIC

```
shallot.FilePropertyDialog.Tab.INDEX_EXOTIC = internalstuff.filepropertydialogtab_index_↩
exotic()  [static]
```

An integer value to be used as base value for specifying order indexes of exotic (i.e.

less important) properties.

1.78.4.3 INDEX_IMPORTANT

```
shallot.FilePropertyDialog.Tab.INDEX_IMPORTANT = internalstuff.filepropertydialogtab_index_↩
important()  [static]
```

An integer value to be used as base value for specifying order indexes of important properties.

1.78.4.4 INDEX_NORMAL

```
shallot.FilePropertyDialog.Tab.INDEX_NORMAL = internalstuff.filepropertydialogtab_index_↩
normal()  [static]
```

An integer value to be used as base value for specifying order indexes of properties with normal importance.

1.78.4.5 INDEX_VERYIMPORTANT

```
shallot.FilePropertyDialog.Tab.INDEX_VERYIMPORTANT = internalstuff.filepropertydialogtab_↩
index_veryimportant()  [static]
```

An integer value to be used as base value for specifying order indexes of very important properties.

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

- /home/pino/projects/shallot/shallot.py

1.79 shallot.FilePropertyDialog.TabPropertyIconTextBanner Class Reference

Represents values for image/text banners as used for shallot.FilePropertyDialog.Tab.PropertyType.IconTextBanner.

Public Member Functions

- def **__init__** (self)
- def add_text (self, text)

  Adds text to the banner.
- def add_icon (self, iconname, size=1.0)

  Adds an icon to the banner.

Public Attributes

- **items**

1.79.1 Detailed Description

Represents values for image/text banners as used for shallot.FilePropertyDialog.Tab.PropertyType.IconTextBanner.

1.79.2 Member Function Documentation

1.79.2.1 add_icon()

```
def shallot.FilePropertyDialog.TabPropertyIconTextBanner.add_icon (
              self,
              iconname,
              size = 1.0 )
```

Adds an icon to the banner.

Parameters

iconname	The name of the icon to append.
size	Icon size (not in pixels but relative to the default).

1.79.2.2 add_text()

```
def shallot.FilePropertyDialog.TabPropertyIconTextBanner.add_text (
              self,
              text )
```

Adds text to the banner.

Parameters

text	The text to append.

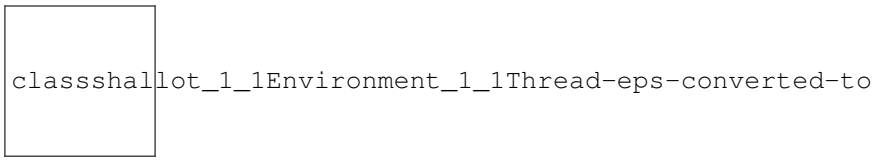
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.80 shallot.Environment.Thread Class Reference

A thread executes code asynchronously.

Inheritance diagram for shallot.Environment.Thread:



Public Member Functions

- def **__init__** (self)
- def run (self)

  Contains the code to execute in this thread.
- def start (self)

  Starts the execution of this thread and calls 'run' in that thread.

Static Public Member Functions

- def execute_threaded (fct, args, kwargs)

  Executes a function in a new shallot.Environment.Thread (and returns that).

1.80.1 Detailed Description

A thread executes code asynchronously.

See also the Python documentation about threading. Although this thread class is not related to the Python thread class (in terms of object orientation), you should understand the general pitfalls which come with multithreading and you might find the Python synchronization tools useful for avoiding those pitfalls. Note: It is strictly forbidden to 'park' a thread and wait for some external event most of the time. Use a shallot.Environment.Timer for recurring tasks.

1.80.2 Member Function Documentation

1.80.2.1 execute_threaded()

```
def shallot.Environment.Thread.execute_threaded (
              fct,
              args,
              kwargs )  [static]
```

Executes a function in a new shallot.Environment.Thread (and returns that).

This avoids subclassing and makes the code more compact. 'args' and 'kwargs' are additional parameters which become the parameters in the actual call of 'fct'.

Parameters

fct	The function to execute.

1.80.2.2 run()

```
def shallot.Environment.Thread.run (
              self )
```

Contains the code to execute in this thread.

Override this method in custom subclasses.

1.80.2.3 start()

```
def shallot.Environment.Thread.start (
              self )
```

Starts the execution of this thread and calls 'run' in that thread.

Returns immediately.

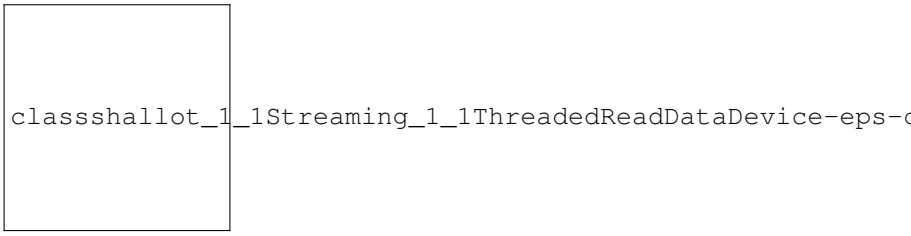
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.81 shallot.Streaming.ThreadedReadDataDevice Class Reference

A binary source dynamically created piecewise in a separate thread.

Inheritance diagram for shallot.Streaming.ThreadedReadDataDevice:



Public Member Functions

- def **__init__** (self)
- def write (self, content)

  Appends a piece of data.
- def run (self)

  This is the content generator function.
- def getdata (self)

  See shallot.Streaming.ReadDataDevice.
- def read (self)

  Reads a chuck of data as Python byte array.
- def readall (self)

  Reads the complete data as Python byte array.

1.81.1 Detailed Description

A binary source dynamically created piecewise in a separate thread.

Override this class and provide the content generator within it.

1.81.2 Member Function Documentation

1.81.2.1 read()

```
def shallot.Streaming.ReadDataDevice.read (
            self )  [inherited]
```

Reads a chuck of data as Python byte array.

An empty array signals the stream ended.

1.81.2.2 run()

```
def shallot.Streaming.ThreadedReadDataDevice.run (
              self )
```

This is the content generator function.

Override this method in custom subclasses. Generate the content in an arbitrary way and call write() once or iteratively until the complete content is written. This method will automatically be executed in a separate thread.

1.81.2.3 write()

```
def shallot.Streaming.ThreadedReadDataDevice.write (
              self,
              content )
```

Appends a piece of data.

Call this function from within the run() method.

Parameters

content	The next piece of content as byte array.

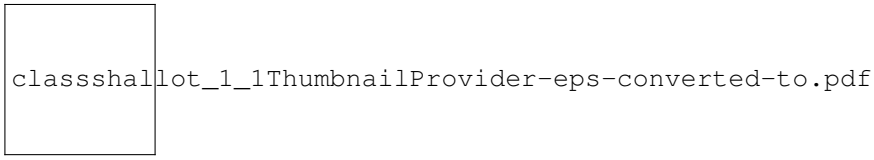
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.82 shallot.ThumbnailProvider Class Reference

Abstract base class for a thumbnail provider.

Inheritance diagram for shallot.ThumbnailProvider:



Public Member Functions

- def **__init__** (self)
- def get_thumbnail (self, operation, node, contenttype, width, height)

  Returns the thumbnail.

Static Public Member Functions

- def register (thumbnailprovider, positionGroup=None, positionIndex=None)

  Registers a custom shallot.ThumbnailProvider instance for offering custom thumbnails.

Static Public Attributes

- INDEX_CORE = internalstuff.thumbnailprovider_index_core()

  Base index value for core (i.e.
- INDEX_NORMAL = internalstuff.thumbnailprovider_index_normal()

  Base index value for thumbnail providers with normal importance.
- INDEX_EXOTIC = internalstuff.thumbnailprovider_index_exotic()

  Base index value for exotic thumbnail providers.
- INDEX_FALLBACK = internalstuff.thumbnailprovider_index_fallback()

  Base index value for fallback thumbnail providers (executed after everything else failed).

1.82.1 Detailed Description

Abstract base class for a thumbnail provider.

It can implement new ways of generating thumbnail images for files. Implement this class and register an instance with shallot.ThumbnailProvider.register.

1.82.2 Member Function Documentation

1.82.2.1 get_thumbnail()

```
def shallot.ThumbnailProvider.get_thumbnail (
                self,
                operation,
                node,
                contenttype,
                width,
                height )
```

Returns the thumbnail.

It must be a bytestring containing the image in a well known format (png, jpeg, svg, ...).

Override this method in custom subclasses.

Parameters

operation	The shallot.Operations.Operation operation object.
node	The shallot.Filesystem.Node filesystem node to get a thumbnail for.
contenttype	The content type of the original file.
width	The requested thumbnail width in pixels.
height	The requested thumbnail height in pixels.

1.82.2.2 register()

```
def shallot.ThumbnailProvider.register (
              thumbnailprovider,
              positionGroup = None,
              positionIndex = None )  [static]
```

Registers a custom shallot.ThumbnailProvider instance for offering custom thumbnails.

Parameters

thumbnailprovider	An instance of shallot.ThumbnailProvider.
positionGroup	Controls execution order. See Position Indexes for details. Use one of the `INDEX_*` values from shallot.ThumbnailProvider.
positionIndex	Controls execution order. See Position Indexes for details.

1.82.3 Member Data Documentation

1.82.3.1 INDEX_CORE

```
shallot.ThumbnailProvider.INDEX_CORE = internalstuff.thumbnailprovider_index_core()  [static]
```

Base index value for core (i.e.

very high priority) thumbnail providers.

1.82.3.2 INDEX_EXOTIC

```
shallot.ThumbnailProvider.INDEX_EXOTIC = internalstuff.thumbnailprovider_index_exotic()  [static]
```

Base index value for exotic thumbnail providers.

1.82.3.3 INDEX_FALLBACK

```
shallot.ThumbnailProvider.INDEX_FALLBACK = internalstuff.thumbnailprovider_index_fallback()
[static]
```

Base index value for fallback thumbnail providers (executed after everything else failed).

1.82.3.4 INDEX_NORMAL

```
shallot.ThumbnailProvider.INDEX_NORMAL = internalstuff.thumbnailprovider_index_normal()  [static]
```

Base index value for thumbnail providers with normal importance.

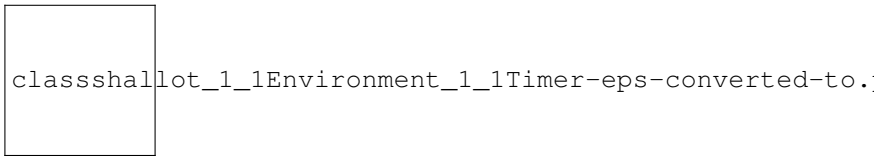
The documentation for this class was generated from the following file:

- /home/pino/projects/shallot/shallot.py

1.83 shallot.Environment.Timer Class Reference

A timers executes some code recurringly in some time interval.

Inheritance diagram for shallot.Environment.Timer:



Public Member Functions

- def **__init__** (self)
- def run (self)

  Contains the code to execute.
- def start (self, interval)

  Starts the timer, which executes 'run' each 'interval' milliseconds.
- def stop (self)

  Stops the timer.
- def is_started (self)

  Returns if the timer is currently active.

Static Public Member Functions

- def start_in_timer (fct, interval, args, kwargs)

  Starts a function in a shallot.Environment.Timer (and returns that).

1.83.1 Detailed Description

A timers executes some code recurringly in some time interval.

1.83.2 Member Function Documentation

1.83.2.1 run()

```
def shallot.Environment.Timer.run (
              self )
```

Contains the code to execute.

Override this method in custom subclasses.

1.83.2.2 start()

```
def shallot.Environment.Timer.start (
              self,
              interval )
```

Starts the timer, which executes 'run' each 'interval' milliseconds.

Returns immediately.

Parameters

interval	The interval in milliseconds.

1.83.2.3 start_in_timer()

```
def shallot.Environment.Timer.start_in_timer (
              fct,
              interval,
              args,
              kwargs )  [static]
```

Starts a function in a shallot.Environment.Timer (and returns that).

This avoids subclassing and makes the code more compact. 'args' and 'kwargs' are additional parameters which become the parameters in the actual call of 'fct'.

Parameters

fct	The function to execute.
interval	The interval in milliseconds.

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

- /home/pino/projects/shallot/shallot.py