Pike Documentation

Release 0.1.1

John Vrbanac

September 24, 2015

Contents

1 Documentation **3**
1.1 User Guide 3
1.2 Classes and Functions 4

2 Indices and tables **7**

Python Module Index **9**

Pike is a dynamic plugin management library for Python. Unlike most Python plugin managers, Pike allows for you to load Python packages from anywhere on a filesystem without complicated configuration. This enables applications to easily add the ability to expand their functionality through plugin modules.

CHAPTER 1

Documentation

1.1 User Guide

1.1.1 Get Started

The common use-case for Pike is to enable dynamic loading of Python packages from various locations on a user's filesystem. This is usually to facilitate the usage of plugins.

The easiest way to use Pike to load Python packages is to use it as a context manager:

```
from pike.manager import PikeManager

with PikeManager(['/path/containing/python/packages']) as mgr:
    classes = mgr.get_classes()
```

If you need to use Pike for an extended period of time (such as for testing), you can use a normal instance of Pike. However, the downside to that is that you'll need to manually trigger Pike to cleanup itself when you're done.

```
from pike.manager import PikeManager

manager = PikeManager(['/path/containing/python/packages'])
classes = manager.get_classes()
manager.cleanup()
```

1.1.2 Discovery

Pike also includes a set of discovery functions to allow for someone to find modules or classes that have been imported or that are available on a filesystem.

- Documentation for imported module discovery: *pike.discovery.py*
- Documentation for filesystem discovery: *pike.discovery.filesystem*

1.1.3 Installation

Install from PyPI

```
pip install --upgrade pike
```

Install from source

You can find the source for Pike located on GitHub. Once downloaded you can install Pike using pip.

If you want to just do a normal source install of Pike the execute:

```
# In the Pike source directory
pip install .
```

If you want to make changes to Pike, then install execute:

```
# In the Pike source directory
pip install -e .
```

1.2 Classes and Functions

1.2.1 Plugin Manager

class `pike.manager.`**`PikeManager`**(*search_paths=None*)

The Pike plugin manager

The manager allows for the dynamic loading of Python packages for any location on a user's filesystem.

Parameters **`search_paths`** (*list*) – List of path strings to include during module importing. These paths are only in addition to existing Python import search paths.

Using PikeManager as a context manager:

```
from pike.manager import PikeManager

with PikeManager(['/path/containing/package']) as mgr:
    import module_in_the_package
```

Using PikeManager instance:

```
from pike.manager import PikeManager

mgr = PikeManager(['/path/container/package'])
import module_in_the_package
mgr.cleanup()
```

`add_to_meta_path`()

Adds Pike's import hooks to Python

This should be automatically handled by Pike; however, this is method is accessible for very rare use-cases.

`cleanup`()

Removes Pike's import hooks

This should be called if an implementer is not using the manager as a context manager.

`get_all_inherited_classes`(*base_class*)

Retrieve all inherited classes from manager's search paths

Parameters **`base_class`** (*Class*) – Base class to filter results by

Returns `List` of all found classes

`get_classes`(*filter_func=None*)

Get all classes within modules on the manager's search paths

Parameters **filter_func** (*Function*) – Custom filter function(cls_obj).

Returns `List` of all found classes

get_module_names()
Get root module names available on the manager's search paths

Returns `generator` providing available module names.

get_package_names()
Get root package names available on the manager's search paths

Returns `generator` providing available package names.

1.2.2 Discovery Classes

Python

pike.discovery.py.**classes_in_module**(*module*, *filter_func=None*)
Retrieve classes within a module

Parameters

- **module** (*module*) – Module to search under
- **filter_func** (*Function*) – Custom filter function(cls_obj).

Returns `generator` containing classes within a module

pike.discovery.py.**get_all_classes**(*module*, *filter_func=None*)
Retrieve all classes from modules

Parameters

- **module** (*module*) – Module to search under
- **filter_func** (*Function*) – Custom filter function(cls_obj).

Returns `List` of all found classes

pike.discovery.py.**get_all_inherited_classes**(*module*, *base_class*)
Retrieve all inherited classes from modules

Parameters

- **module** (*module*) – Module to search under
- **base_class** (*Class*) – Base class to filter results by

Returns `List` of all found classes

pike.discovery.py.**get_child_modules**(*module*, *recursive=True*)
Retrieve child modules

Parameters

- **module** (*module*) – Module to search under
- **recursive** (*bool*) – Toggles the retrieval of sub-children module.

Returns `generator` containing child modules

pike.discovery.py.**get_inherited_classes**(*module*, *base_class*)
Retrieve inherited classes from a single module

Parameters

- **module** (*module*) – Module to search under
- **base_class** (*Class*) – Base class to filter results by

Returns `List` of all found classes

`pike.discovery.py.`**`get_module_by_name`**(*full_module_name*)
Import module by full name

Parameters **full_module_name** (*str*) – Full module name e.g. (pike.discovery.py)

Returns Imported `module`

Filesystem

`pike.discovery.filesystem.`**`find_modules`**(*path*)
Finds all modules located on a path

`pike.discovery.filesystem.`**`find_packages`**(*path*)
Finds all packages located on a path

`pike.discovery.filesystem.`**`is_module`**(*path*)
Checks if path string is a module

`pike.discovery.filesystem.`**`is_package`**(*path*)
Checks if path string is a package

`pike.discovery.filesystem.`**`recursive_find_modules`**(*path*)
Recursively finds all modules located on a path

`pike.discovery.filesystem.`**`recursive_find_packages`**(*path*)
Recursively finds all packages located on a path

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

p

pike.discovery.filesystem, 6
pike.discovery.py, 5

A

add_to_meta_path() (pike.manager.PikeManager method), 4

C

classes_in_module() (in module pike.discovery.py), 5
cleanup() (pike.manager.PikeManager method), 4

F

find_modules() (in module pike.discovery.filesystem), 6
find_packages() (in module pike.discovery.filesystem), 6

G

get_all_classes() (in module pike.discovery.py), 5
get_all_inherited_classes() (in module pike.discovery.py), 5
get_all_inherited_classes() (pike.manager.PikeManager method), 4
get_child_modules() (in module pike.discovery.py), 5
get_classes() (pike.manager.PikeManager method), 4
get_inherited_classes() (in module pike.discovery.py), 5
get_module_by_name() (in module pike.discovery.py), 6
get_module_names() (pike.manager.PikeManager method), 5
get_package_names() (pike.manager.PikeManager method), 5

I

is_module() (in module pike.discovery.filesystem), 6
is_package() (in module pike.discovery.filesystem), 6

P

pike.discovery.filesystem (module), 6
pike.discovery.py (module), 5
PikeManager (class in pike.manager), 4

R

recursive_find_modules() (in module pike.discovery.filesystem), 6
recursive_find_packages() (in module pike.discovery.filesystem), 6