loggers Documentation

Release 0.1.4

Jonatan Dellagostin

Oct 02, 2018

Contents

CHAPTER 1

loggers

loggers is a Python library that provides usefull wrapper methods for logging class. To be used as a superclass for your own classes.

1.1 Example

```
>>> from loggers import Loggers
>>>
>>> class spamClass(Loggers):
...   def __init__(self, log_folder=None):
...       super(spamClass, self).__init__('spamClass', log_folder_path=log_folder)
...   def do_stuff(self, arg):
...       if not type(arg) == str:
...         self.log.error("I was expecting a string. :( ")
...       else:
...         self.log.debug("I received my string. :)")
...
>>> spam = spamClass('/tmp/logs/spamClass')
>>> spam.log.error('ERROR')
Log: ERROR | Log level:ERROR | Date:31/10/2016 16:51:47
>>> spam.set_log_rotate_handler(True)
>>> spam.do_stuff(123)
Log: I was expecting a string. :(  | Log level:ERROR | Date:31/10/2016 16:51:47
>>> spam.do_stuff('Eggs')
>>> spam.set_log_level('DEBUG')
Log: Changing log level to DEBUG | Log level:DEBUG | Date:31/10/2016 16:51:47
>>> spam.do_stuff('Spam')
Log: I received my string. :) | Log level:DEBUG | Date:31/10/2016 16:51:47
```

1.2 Installation

To install loggers, simply run:

```
$ pip install loggers
```

loggers is compatible with Python 2.6+ and Python 3

1.3 Documentation

https://loggers.readthedocs.io

1.4 Source Code

Feel free to fork, evaluate and contribute to this project.

Source: https://github.com/jonDel/loggers

1.5 License

GPLv3 licensed.

CHAPTER 2

loggers package contents:

2.1 loggers package

2.1.1 Submodules

2.1.2 loggers.loggers module

class `loggers.loggers.`**`Loggers`**(*log_name*, ***kwargs*)

Bases: `object`

Provides log functionalities either in stream or file form

Parameters

- **`log_name`** (`str`) – name of the log handler
- **`log_folder_path`** (`str`,optional, *default* =None) – folder where the log's files will lie
- **`log_file`** (`str`,optional, *default* =None) – path of the debug and error log's files
- **`logger`** (`obj`,optional, *default* =None) – use this preexistent logger instead of creating a new one

`set_log_format`(*log_type*, *log_format*)

Configures log format

Parameters

- **`log_type`** (`str`) – log type (error, debug or stream)
- **`log_format`** (`str`) – log format (ex:"Log: %(message)s | Log level:%(levelname)s | Date:%(asctime)s',datefmt='%m/%d/%Y %I:%M:%S")

`set_log_level`(*log_level*)

Configures class log level

Parameters **`log_level`** (`str`) – log level ('NOTSET','DEBUG','INFO' 'WARNING', 'ERROR', 'CRITICAL')

`set_log_rotate_handler`(*set_file*)

Enables/disables logs to be written to files

Parameters **`set_file`** (`bool`) – False disables, True enables

2.1.3 Module contents

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index