

---

# **BioMaj Manager Documentation**

***Release 1.0.3***

**Emmanuel Quevillon**

August 23, 2016



<b>1</b>	<b>decorators API reference</b>	<b>3</b>
<b>2</b>	<b>links API reference</b>	<b>5</b>
<b>3</b>	<b>manager API reference</b>	<b>7</b>
<b>4</b>	<b>news API reference</b>	<b>9</b>
<b>5</b>	<b>plugins API reference</b>	<b>11</b>
<b>6</b>	<b>utils API reference</b>	<b>13</b>
<b>7</b>	<b>writer API reference</b>	<b>17</b>
<b>8</b>	<b>Indices and tables</b>	<b>19</b>
	<b>Python Module Index</b>	<b>21</b>



Contents:



---

## decorators API reference

---

Global decorators for BioMAJ Manager

`biomajmanager.decorators.bank_required(func)`

Decorator function that checks a bank name is set

**Parameters** `func` (*Function*) – Decorated function

**Returns** Result of function called

**Return type** `func`

**Raises** `SystemExit` – If no bank set

`biomajmanager.decorators.deprecated(func)`

This is a decorator which can be used to mark functions as deprecated.

It will result in a warning being emitted when the function is used.

**Parameters** `func` (*Function*) – Decorated function

**Returns** Result of function called

**Return type** `func`

`biomajmanager.decorators.user_granted(func)`

Decorator function that checks a user has enough right to perform action

**Parameters** `func` (*Function*) – Decorated function

**Returns** Result of function called

**Return type** `func`

**Raises** `SystemExit` – If no owner found either in bank nor in config file





---

**links API reference**

---



---

## manager API reference

---

**Manager.SAVE\_BANK\_LINE\_PATTERN:** Print pattern (str), %-20s\t%-30s\t%-20s\t%-20s\t%-20s\n

**Manager.simulate:** Simulate mode (bool), False

**Manager.verbose:** Verbose mode (bool), False



---

## news API reference

---

`News.MAX_NEWS`: Maximum number of news returned while reading news directory (int), 5



---

## plugins API reference

---

Plugins mechanism to load user defined method

**class** `biomajmanager.plugins.BMPlugin`  
Base plugin class for BioMAJ manager

**get\_config()**  
Get the BioMAJ manager config as object

**Returns** configparser instance

**Return type** :class:'configparser'

**get\_manager()**  
Get the BioMAJ manager instance

**Returns** Manager

**Return type** `biomaj.manager.Manager`

**get\_name()**  
Get the name of the plugin. Based on the class name

**set\_config(config)**  
Set BioMAJ manager config object

**set\_manager(manager)**  
Set BioMAJ manager config object

**class** `biomajmanager.plugins.Plugins` (*manager=None, name=None*)  
Plugin class for BioMAJ Manager

**\_\_init\_\_** (*manager=None, name=None*)  
Create the plugin object

### Parameters

- **manager** (`biomajmanager.manager.Manager`) – Manager instance
- **name** (*String*) – Name of the plugin to load. [DEFAULT: load all plugins]

### Raises

- **SystemExit** – If 'manager' arg is not given
- **SystemExit** – If 'PLUGINS' section not found in `manager.properties`
- **SystemExit** – If 'plugins.dir' not set in `manager.properties`
- **SystemExit** – If 'plugins.list' not set in `manager.properties`

- **SystemExit** – If ‘plugins.dir’ does not exist

**\_\_weakref\_\_**

list of weak references to the object (if defined)



---

## utils API reference

---

**Utils.DATE\_FMT:** date format pattern (*str*), %Y-%m-%d %H:%M:%S

**Utils.show\_warn:** show warnings mesasge (*bool*), True

**Utils.show\_debug:** show debug message (*bool*), True

**Utils.show\_verbose:** show verbose message (*bool*), True

**Utils.timer\_start:** Start timer value (*float*), 0.0

**Utils.timer\_stop:** Stop timer value (*float*), 0.0

Utilities class for BioMAJ Manager

**class** `biomajmanager.utils.Utils`

Utility class

**\_\_weakref\_\_**

list of weak references to the object (if defined)

**static** `_print` (*msg*, *to*=<open file '<stdout>', mode 'w'>)

Redefined print function to support python 2 and 3

### Parameters

- **msg** (*str*) – Message to print
- **to** (*file*) – File handle

**Returns** Message to print

**Return type** *str*

**static** `clean_symlinks` (*path*=None, *delete*=False)

Search for broken symlinks.

Given a path, it search for all symlinks 'path' directory and remove broken symlinks. If delete is True, remove broken symlink(s), otherwise lists broken symlinks.

### Parameters

- **path** (*str*) – Path to search symlinks from
- **delete** (*bool*) – Wether to delete or not broken symlink(s)

**Returns** Number of cleaned/deleted link(s)

**Return type** *int*

**Raises**

- **SystemExit** – If path not found, or not given
- **SystemExit** – If remove of symlink(s) failed

**static elapsed\_time** ()

Get the elapsed time between start and stop timer.

Stop timer call is not required. If not set, it is automatically called as soon as the method is called

**Returns** Elapsed time

**Return type** `float`

**Raises** **SystemExit** – If `Utils.timer_start` is not defined

**static error** (*msg*)

Prints error message on `STDERR` and exits with exit code 1

**Parameters** **msg** (*str*) – Message to print

**Returns** Error message

**Return type** `str`

**Raises** **SystemExit** –

**static get\_broken\_links** (*path=None*)

Search for broken symlinks from a particular path.

If path is not given or `None`, then it search from the production directory.

**Parameters** **path** (*str*) – Path to search broken links from

**Returns** Number of found broken links

**Return type** `int`

**Raises** **SystemExit** – If path does not exist

**static get\_deepest\_dir** (*path=None, full=False*)

Return only one deepest dir from the path

**Parameters**

- **path** (*str*) – Path
- **full** (*bool*) – Returns complete path or not

**Returns** Directory name

**Return type** `str`

**static get\_deepest\_dirs** (*path=None, full=False*)

Get the last directories from a path

**Parameters**

- **path** (*str*) – Path to start from
- **full** (*bool*) – Get the full path otherwise the directory only

**Returns** List of directories

**Return type** `list`

**Raises** **SystemExit** – If 'path' not given or does not exist

**static get\_files** (*path=None*)

Return the list of file(s) found for a given path

**Parameters** `path (str)` – Path to search from

**Returns** List of file(s) found

**Return type** `list`

**Raises** `SystemExit` – If path does not exist

**static** `get_now()`

Get current time from `time.time` formatted using `Utils.DATE_FMT`

**Returns** Current time formatted using `Utils.DATE_FMT`

**Return type** `time.time`

**static** `get_subtree(path=None)`

Get the subtree structure from a root path

E.g.: File system is `/t/a1/a2/a3`, `get_subtree(path='/t')` -> `/a1/a2/a3` :param path: Root path to get subtree structure from :type path: str :return: List of found subtree :rtype: list

**static** `ok(msg)`

Prints a [OK] msg

**Parameters** `msg (str)` – Message to print

**Returns** Message to print

**Return type** `str`

**static** `reset_timer()`

Reset to `0.0` `timer_start()` and `timer_stop()` for a new `elapsed_time()` count

**static** `start_timer()`

Set current time at function call

**static** `stop_timer()`

Set current time at function call

**static** `time2date(otime)`

Convert a timestamp into a datetime object

**Parameters** `otime (time)` – Timestamp to convert

**Returns** Formatted time to date

**Return type** `datetime.datetime`

**static** `time2datefmt(otime,fmt='%Y-%m-%d %H:%M:%S')`

Converts a timestamp into a date following the format `fmt`, default to `Utils.DATE_FMT`

**Parameters**

- `otime (time)` – Timestamp to convert
- `fmt (str)` – Date format to follow for conversion

**Returns** Formatted time to date

**Return type** `datetime.datetime`

**static** `user()`

Returns the current user running or using the script. Taken from `os.env`

**Returns** User name

**Return type** `str`

**static verbose** (*msg*)

Prints verbose message. Requires Manager.verbose to be True

**Parameters** **msg** (*str*) – Verbose message to print

**Returns** Verbose message

**Return type** *str*

**static warn** (*msg*)

Prints warning message. Required Utils.show\_warn to be set to True

**Parameters** **msg** (*str*) – Warning message to print

**Returns** Warning message

**Return type** *str*

---

## writer API reference

---

Writer class to be used with Jinja2 templates

**class** `biomajmanager.writer.Writer` (*template\_dir=None, config=None, output=None*)

Writer class for BioMAJ manager to create what's desired as output

**\_\_init\_\_** (*template\_dir=None, config=None, output=None*)

Create Writer object

### Parameters

- **template\_dir** (*str*) – Root directory where to find templates
- **config** (*configparser*) – Global configuration file from BiomajConfig
- **output** (*str*) – Output file. Default STDOUT

### Raises

- **SystemExit** – If 'template\_dir' is not given
- **SystemExit** – If 'MANAGER' section not found in `manager.properties`
- **SystemExit** – If 'template.dir' not set in `manager.properties`

**\_\_weakref\_\_**

list of weak references to the object (if defined)

**write** (*template=None, data=None*)

Print template 'data' to stdout using template file 'template'.

'data' arg can be left None, this way method can be used to render file from scratch

### Parameters

- **template** (*str*) – Template file name
- **data** (*dict*) – Template data

**Returns** True, throws on error

**Return type** `bool`

### Raises

- **SystemExit** – If 'template' is None
- **SystemExit** – If 'template' is not found
- **SystemExit** – If 'template' has a syntax error in it
- **SystemExit** – If 'output' file cannot be opened



---

## Indices and tables

---

- `genindex`
- `modindex`
- `search`





## **b**

`biomajmanager.decorators`, [3](#)  
`biomajmanager.plugins`, [11](#)  
`biomajmanager.utils`, [13](#)  
`biomajmanager.writer`, [17](#)



## Symbols

`__init__()` (biomajmanager.plugins.Plugins method), 11  
`__init__()` (biomajmanager.writer.Writer method), 17  
`__weakref__` (biomajmanager.plugins.Plugins attribute), 12  
`__weakref__` (biomajmanager.utils.Utils attribute), 13  
`__weakref__` (biomajmanager.writer.Writer attribute), 17  
`_print()` (biomajmanager.utils.Utils static method), 13

## B

`bank_required()` (in module biomajmanager.decorators), 3  
biomajmanager.decorators (module), 3  
biomajmanager.plugins (module), 11  
biomajmanager.utils (module), 13  
biomajmanager.writer (module), 17  
BMPlugin (class in biomajmanager.plugins), 11

## C

`clean_symlinks()` (biomajmanager.utils.Utils static method), 13

## D

`deprecated()` (in module biomajmanager.decorators), 3

## E

`elapsed_time()` (biomajmanager.utils.Utils static method), 14  
`error()` (biomajmanager.utils.Utils static method), 14

## G

`get_broken_links()` (biomajmanager.utils.Utils static method), 14  
`get_config()` (biomajmanager.plugins.BMPlugin method), 11  
`get_deepest_dir()` (biomajmanager.utils.Utils static method), 14  
`get_deepest_dirs()` (biomajmanager.utils.Utils static method), 14  
`get_files()` (biomajmanager.utils.Utils static method), 14

`get_manager()` (biomajmanager.plugins.BMPlugin method), 11  
`get_name()` (biomajmanager.plugins.BMPlugin method), 11  
`get_now()` (biomajmanager.utils.Utils static method), 15  
`get_subtree()` (biomajmanager.utils.Utils static method), 15

## O

`ok()` (biomajmanager.utils.Utils static method), 15

## P

Plugins (class in biomajmanager.plugins), 11

## R

`reset_timer()` (biomajmanager.utils.Utils static method), 15

## S

`set_config()` (biomajmanager.plugins.BMPlugin method), 11  
`set_manager()` (biomajmanager.plugins.BMPlugin method), 11  
`start_timer()` (biomajmanager.utils.Utils static method), 15  
`stop_timer()` (biomajmanager.utils.Utils static method), 15

## T

`time2date()` (biomajmanager.utils.Utils static method), 15  
`time2datefmt()` (biomajmanager.utils.Utils static method), 15

## U

`user()` (biomajmanager.utils.Utils static method), 15  
`user_granted()` (in module biomajmanager.decorators), 3  
Utils (class in biomajmanager.utils), 13

## V

`verbose()` (biomajmanager.utils.Utils static method), 15

## W

`warn()` (biomajmanager.utils.Utils static method), [16](#)

`write()` (biomajmanager.writer.Writer method), [17](#)

`Writer` (class in biomajmanager.writer), [17](#)