

---

# Mirrors Documentation

*Release 0.4.0*

**Ethan House**

December 26, 2016



<b>1</b>	<b>Getting Started</b>	<b>3</b>
1.1	Installation . . . . .	3
1.2	Configuration . . . . .	3
<b>2</b>	<b>Usage</b>	<b>5</b>
<b>3</b>	<b>Source Docs</b>	<b>7</b>
3.1	repo.py . . . . .	7
3.2	libmirrors.py . . . . .	9
3.3	libcmd.py . . . . .	9
<b>4</b>	<b>Indices and tables</b>	<b>11</b>



Contents:



---

## Getting Started

---

### 1.1 Installation

#### 1.1.1 Pre-Build

```
git clone http://github.com/ehouse/mirrors.git
./setup.py build
```

#### 1.1.2 Install

Running in a virtualenv isn't required but recommended.

```
./setup.py install
```

#### 1.1.3 Uninstall

```
pip uninstall mirrors
```

### 1.2 Configuration

Configuration can be in any file you choose however mirrors.conf is what is included. The file must be included when mirrors is run with the -c flag.

(ex. mirrors -c mirrors.conf)

#### 1.2.1 Global Config Options

[GLOBAL]

All of the global configuration is stored within the [GLOBAL] block. If it is not present the application will fail to load.

```
async_processes = 4
```

The number of syncs that may be run at any given time. Scheduled syncs will wait until a spot opens up before it will begin running.

```
log_file = ./mirrors.log
```

Application log file. This is where errors, warnings or general information is logged to. Default is mirrors.log

```
check_sleep = 30
```

Time to sleep in between syncs . Default is 30 seconds.

### 1.2.2 Repo Options

```
[test1]
```

Name of the repo in brackets. This is how it will be accessed within the repl.

```
source = rsync://mirrors.rit.edu/FreeBSD
```

Source of the rsync transfer.

```
rsync_args = -avhz
```

Flags to pass into the rsync process.

```
destination = ./distros/
```

Destination of the rsync file transfer.

```
weight = 0
```

Weight of the sync. Lower numbers will go before higher numbers. Value between -10 and 10. Default is 0

```
pre_command =
```

Shell command to run before the rsync starts.

```
post_command =
```

Shell command to run after the rsync finishes.

```
log_file = ./log/LDP.log
```

Location of the repo log file. Rsync STDOUT and STDERR are piped here.

```
async_sleep = 2h
```

Time to wait after a sync has completed before it is re-queued.

```
hourly_sync = 0,6.5,12,18.5
```

Strict time frame for syncs to run.



---

**Usage**

---



### 3.1 repo.py

`class mirrors.repo.Repo(name, config)`

`__init__(name, config)`

A repo object which stores info about a single repo.

**Parameters**

- **name** (*str*) – Name of repo
- **config** (*ConfigParser.ConfigParser*) – running config options

`__weakref__`

list of weak references to the object (if defined)

`is_alive()`

Bool of syncing status.

`kill()`

Send SIGKILL To the rsync process.

`class rsync_thread(name, config)`

Extended threading.Thread class to control rsync via subprocess.

**Parameters**

- **name** (*str*) – Name of repo
- **config** (*Configparser.Configparser*) – Running config options

`Repo.running_time()`

Total running time of active sync.

**Return type** int

**Returns** An int of total syncing time elapsed

**Return type** None

**Returns** None if not syncing

`Repo.sleep_time()`

Sleep duration of sleeping sync.

**Return type** int

**Returns** A int of time elapsed since sleeping

**Return type** None

**Returns** None if not in sleeping state

`Repo.start_sync()`

Run an rsync against the repo source.

`Repo.terminate()`

Send SIGTERM To the rsync process.

`Repo.time_remaining()`

Return time left until sleep is over.

**Return type** int

**Returns** A int of time remaining in sleep state

**Return type** None

**Returns** None if not in sleeping state

`class mirrors.repo.RepoManager (config)`

`__init__ (config)`

Singleton manager of the repositories and threading.

**Parameters** `config` (*Configparser.Configparser*) – Running config options

`__metaclass__`

alias of Singleton

`__weakref__`

list of weak references to the object (if defined)

`activate (name)`

Activate repo for syncing.

**Parameters** `name` (*str*) – Name of Repo

**Raises** `Repo.RepoError` if no repo exists by given name

`add_repo (name)`

Create a repo for a section in the running config.

**Parameters** `name` (*str*) – Name of repo

**Raises** `Repo.RepoConfigError` if no config exists for given repo name

`deactivate (name)`

Deactivate repo from syncing.

**Parameters** `name` (*str*) – Name of repo

**Raises** `Repo.RepoError` if no repo exists by given name

`del_repo (name)`

Delete repo object from dict.

**Parameters** `name` (*str*) – Name of repo

**Raises** `Repo.RepoError` if no repo exists by passed in name.

`enqueue (name)`

Add repo to the queue.

**Parameters** `name` (*str*) – Name of repo

**Raises** `Repo.RepoError` if repo is already queued or doesn't exist

**gen\_repo** ()

Generator for `repo_dict`.

**Return type** `Repo`

**Returns** `Repo` Object

**get\_repo** (*name*)

Return repo object if exists.

**Parameters** `name` (*str*) – name of repo

**Return type** `Repo`

**Returns** `Repo` Object

**Return type** `None`

**Returns** `None` if no repo exists by passed in name

**status** (*name*)

Return status of `Repo`.

**Parameters** `name` (*str*) – Name of `Repo`

**Return type** `str`

**Returns** `str` status of `Repo`

## 3.2 libmirrors.py

`mirrors.libmirrors.t2s` (*s*)

Converts human readable time to seconds.

**Parameters** `s` (*str*) – Human readable time string (ex. 5m or 2h)

**Return type** `int`

**Returns** `int` time converted to seconds

## 3.3 libcmd.py

`class mirrors.cmdline.Console` (*repo\_manager*)

**do\_activate** (*\*args*)

Activate repo for syncing.

**do\_add** (*\*args*)

Add repo from running config.

TODO

**do\_config** (*\*args*)

Edit configuration running settings.

TODO

**do\_deactivate** (\*args)  
Deactivate repo from syncing.

**do\_del** (\*args)  
Delete repo from running config.  
TODO

**do\_enqueue** (\*args)  
Add repo onto the end of the async queue.

**do\_exit** (\*args)  
Stops all syncs and terminates mirrors.

**do\_forcekill** (\*args)  
Send SIGKILL to rsync process.

**do\_kill** (\*args)  
Send SIGTERM to rsync process.

**do\_list** (\*args)  
List all of the loaded repos.

**do\_print** (\*args)  
Print config and status events.  
TODO

**do\_quit** (\*args)  
Stops all syncs and terminates mirrors.

**do\_reload** (\*args)  
Reload either individual or entire config.  
TODO

**do\_start** (\*args)  
Add repo onto the end of the async queue.

**do\_status** (\*args)  
Prints status table or the status of an individual sync.

**do\_terminate** (\*args)  
Send SIGTERM to rsync process.

**do\_write** (\*args)  
Write configuration settings to file.  
TODO

**postloop** ()  
postloop cleanup.

---

## Indices and tables

---

- *genindex*
- *search*





## Symbols

`__init__()` (mirrors.repo.Repo method), 7  
`__init__()` (mirrors.repo.RepoManager method), 8  
`__metaclass__` (mirrors.repo.RepoManager attribute), 8  
`__weakref__` (mirrors.repo.Repo attribute), 7  
`__weakref__` (mirrors.repo.RepoManager attribute), 8

## A

`activate()` (mirrors.repo.RepoManager method), 8  
`add_repo()` (mirrors.repo.RepoManager method), 8

## C

Console (class in mirrors.cmdline), 9

## D

`deactivate()` (mirrors.repo.RepoManager method), 8  
`del_repo()` (mirrors.repo.RepoManager method), 8  
`do_activate()` (mirrors.cmdline.Console method), 9  
`do_add()` (mirrors.cmdline.Console method), 9  
`do_config()` (mirrors.cmdline.Console method), 9  
`do_deactivate()` (mirrors.cmdline.Console method), 9  
`do_del()` (mirrors.cmdline.Console method), 10  
`do_enqueue()` (mirrors.cmdline.Console method), 10  
`do_exit()` (mirrors.cmdline.Console method), 10  
`do_forcekill()` (mirrors.cmdline.Console method), 10  
`do_kill()` (mirrors.cmdline.Console method), 10  
`do_list()` (mirrors.cmdline.Console method), 10  
`do_print()` (mirrors.cmdline.Console method), 10  
`do_quit()` (mirrors.cmdline.Console method), 10  
`do_reload()` (mirrors.cmdline.Console method), 10  
`do_start()` (mirrors.cmdline.Console method), 10  
`do_status()` (mirrors.cmdline.Console method), 10  
`do_terminate()` (mirrors.cmdline.Console method), 10  
`do_write()` (mirrors.cmdline.Console method), 10

## E

`enqueue()` (mirrors.repo.RepoManager method), 8

## G

`gen_repo()` (mirrors.repo.RepoManager method), 9

`get_repo()` (mirrors.repo.RepoManager method), 9

## I

`is_alive()` (mirrors.repo.Repo method), 7

## K

`kill()` (mirrors.repo.Repo method), 7

## P

`postloop()` (mirrors.cmdline.Console method), 10

## R

Repo (class in mirrors.repo), 7  
Repo.rsync\_thread (class in mirrors.repo), 7  
RepoManager (class in mirrors.repo), 8  
`running_time()` (mirrors.repo.Repo method), 7

## S

`sleep_time()` (mirrors.repo.Repo method), 7  
`start_sync()` (mirrors.repo.Repo method), 8  
`status()` (mirrors.repo.RepoManager method), 9

## T

`t2s()` (in module mirrors.libmirrors), 9  
`terminate()` (mirrors.repo.Repo method), 8  
`time_remaining()` (mirrors.repo.Repo method), 8