Python LiveReload Documentation

Release 0.3

Hsiaoming Yang

Nov 20, 2018

Contents

This is a brand new LiveReload in version 2.0.0.

Download on PyPi

CHAPTER 1

Installation

Python LiveReload is designed for web developers who know Python.

Install Python LiveReload with pip:

```
$ pip install livereload
```

If you don't have pip installed, try easy_install:

```
$ easy_install livereload
```

CHAPTER 2

Command Line Interface

Python LiveReload provides a command line utility, `livereload`, for starting a server in a directory.

By default, it will listen to port 35729, the common port for LiveReload browser extensions.

```
$ livereload --help
usage: livereload [-h] [-p PORT] [-w WAIT] [directory]

Start a `livereload` server

positional arguments:
  directory             Directory to watch for changes

optional arguments:
  -h, --help            show this help message and exit
  -p PORT, --port PORT  Port to run `livereload` server on
  -w WAIT, --wait WAIT  Time delay before reloading
```

Older versions of Python LiveReload used a `Guardfile` to describe optional additional rules for files to watch and build commands to run on changes. This conflicted with other tools that used the same file for their configuration and is no longer supported since Python LiveReload version 2.0.0. Instead of a `Guardfile` you can now write a Python script using very similar syntax and run it instead of the command line application.

CHAPTER 3

Script example: Sphinx

Here's a simple example script that rebuilds Sphinx documentation:

```
#!/usr/bin/env python
from livereload import Server, shell
server = Server()
server.watch('docs/*.rst', shell('make html', cwd='docs'))
server.serve(root='docs/_build/html')
```

Run it, then open http://localhost:5500/ and you can see the documentation changes in real time.

CHAPTER 4

Developer Guide

The new livereload server is designed for developers. It can power a wsgi application now:

```
from livereload import Server, shell

server = Server(wsgi_app)

# run a shell command
server.watch('static/*.stylus', 'make static')

# run a function
def alert():
    print('foo')
server.watch('foo.txt', alert)

# output stdout into a file
server.watch('style.less', shell('lessc style.less', output='style.css'))

server.serve()
```

The `Server` class accepts parameters:

- app: a wsgi application
- watcher: a watcher instance, you don't have to create one

4.1 server.watch

`server.watch` can watch a filepath, a directory and a glob pattern:

```
server.watch('path/to/file.txt')
server.watch('directory/path/')
server.watch('glob/*.pattern')
```

You can also use other library (for example: formic) for more powerful file adding:

```
for filepath in formic.FileSet(include="**.css"):
    server.watch(filepath, 'make css')
```

You can delay a certain seconds to send the reload signal:

```
# delay 2 seconds for reloading
server.watch('path/to/file', delay=2)
```

4.2 server.serve

Setup a server with `server.serve` method. It can create a static server and a livereload server:

```
# use default settings
server.serve()

# livereload on another port
server.serve(liveport=35729)

# use custom host and port
server.serve(port=8080, host='localhost')

# open the web browser on startup, based on $BROWSER environment variable
server.serve(open_url_delay=5, debug=False)
```

4.3 shell

The powerful `shell` function will help you to execute shell commands. You can use it with `server.watch`:

```
# you can redirect command output to a file
server.watch('style.less', shell('lessc style.less', output='style.css'))

# commands can be a list
server.watch('style.less', shell(['lessc', 'style.less'], output='style.css'))

# working with Makefile
server.watch('assets/*.styl', shell('make assets', cwd='assets'))
```

CHAPTER 5

Frameworks Integration

Livereload can work seamlessly with your favorite framework.

5.1 Django

For Django there is a management command included.

To use simply

- add `'livereload'` to your `INSTALLED_APPS` and
- then run `./manage.py livereload`.

For available options like host and ports please refer to `./manage.py livereload -h`.

To automagically serve static files like the native `runserver` command you have to use dj-static. (follow the simple instructions there).

5.2 Flask

Wrap Flask with livereload is much simpler:

```
# app is a Flask object
app = create_app()

# remember to use DEBUG mode for templates auto reload
# https://github.com/lepture/python-livereload/issues/144
app.debug = True

server = Server(app.wsgi_app)
# server.watch
server.serve()
```

5.3 Bottle

Wrap the `Bottle` app with livereload server:

```
# Without this line templates won't auto reload because of caching.
# http://bottlepy.org/docs/dev/tutorial.html#templates
bottle.debug(True)

app = Bottle()
server = Server(app)
# server.watch
server.serve()
```

CHAPTER 6

API

class livereload.**Server**(*app=None*, *watcher=None*)

Livereload server interface.

Initialize a server and watch file changes:

```
server = Server(wsgi_app)
server.serve()
```

Parameters

- **app** – a wsgi application instance
- **watcher** – A Watcher instance, you don't have to initialize it by yourself. Under Linux, you will want to install pyinotify and use INotifyWatcher() to avoid wasted CPU usage.

serve(*port=5500*, *liveport=None*, *host=None*, *root=None*, *debug=None*, *open_url=False*, *restart_delay=2*, *open_url_delay=None*, *live_css=True*)

Start serve the server with the given port.

Parameters

- **port** – serve on this port, default is 5500
- **liveport** – live reload on this port
- **host** – serve on this hostname, default is 127.0.0.1
- **root** – serve static on this root directory
- **debug** – set debug mode, which autoreloads the app on code changes via Tornado (and causes polling). Defaults to True when `self.app` is set, otherwise False.
- **open_url_delay** – open webbrowser after the delay seconds
- **live_css** – whether to use live css or force reload on css. Defaults to True

watch(*filepath*, *func=None*, *delay=None*, *ignore=None*)

Add the given filepath for watcher list.

Once you have intialized a server, watch file changes before serve the server:

```
server.watch('static/*.stylus', 'make static')
def alert():
    print('foo')
server.watch('foo.txt', alert)
server.serve()
```

Parameters

- **filepath** – files to be watched, it can be a filepath, a directory, or a glob pattern
- **func** – the function to be called, it can be a string of shell command, or any callable object without parameters
- **delay** – Delay sending the reload message. Use 'forever' to not send it. This is useful to compile sass files to css, but reload on changed css files then only.
- **ignore** – A function return True to ignore a certain pattern of filepath.

livereload.**shell**(*cmd*, *output=None*, *mode='w'*, *cwd=None*, *shell=False*)

Execute a shell command.

You can add a shell command:

```
server.watch(
    'style.less', shell('lessc style.less', output='style.css')
)
```

Parameters

- **cmd** – a shell command, string or list
- **output** – output stdout to the given file
- **mode** – only works with output, mode `w` means write, mode `a` means append
- **cwd** – set working directory before command is executed.
- **shell** – if true, on Unix the executable argument specifies a replacement shell for the default `/bin/sh`.

CHAPTER 7

Changelog

The full list of changes between each Python LiveReload release.

7.1 Changelog

The full list of changes between each Python LiveReload release.

7.1.1 Version 2.6.0

Released on Nov 21, 2018

1. Changed logic of liveport.
2. Fixed bugs

7.1.2 Version 2.5.2

Released on May 2, 2018

1. Fix tornado 4.5+ not closing connection
2. Add ignore dirs
3. Fix bugs

7.1.3 Version 2.5.1

Release on Jan 7, 2017

Happy New Year.

1. Fix Content-Type detection

2. Ensure current version of pyinotify is installed before using

7.1.4 Version 2.5.0

Released on Nov 16, 2016

1. wait parameter can be float via Todd Wolfson
2. Option to disable liveCSS via Yunchi Luo
3. Django management command via Marc-Stefan Cassola

7.1.5 Version 2.4.1

Released on Jan 19, 2016

1. Allow other hostname with JS script location.hostname
2. Expose delay parameter in command line tool
3. Server.watch accept ignore parameter

7.1.6 Version 2.4.0

Released on May 29, 2015

1. Fix unicode issue with tornado built-in StaticFileHandler
2. Add filter for directory watching
3. Watch without browser open
4. Auto use inotify wather if possible
5. Add `open_url_delay` parameter
6. Refactor lots of code.

Thanks for the patches and issues from everyone.

7.1.7 Version 2.3.2

Released on Nov 5, 2014

1. Fix root parameter in `serve` method via #76.
2. Fix shell unicode stdout error.
3. More useful documentation.

7.1.8 Version 2.3.1

Released on Nov 1, 2014

1. Add `cwd` parameter for `shell`
2. When `delay` is `forever`, it will not trigger a livereload
3. Support different ports for app and livereload.

7.1.9 Version 2.3.0

Released on Oct 28, 2014

1. Add '–host' argument to CLI
2. Autoreload when python code changed
3. Add delay parameter to watcher

7.1.10 Version 2.2.2

Released on Sep 10, 2014

Fix for tornado 4.

7.1.11 Version 2.2.1

Released on Jul 10, 2014

Fix for Python 3.x

7.1.12 Version 2.2.0

Released on Mar 15, 2014

- Add bin/livereload
- Add inotify support

7.1.13 Version 2.1.0

Released on Jan 26, 2014

Add ForceReloadHandler.

7.1.14 Version 2.0.0

Released on Dec 30, 2013

A new designed livereload server which has the power to serve a wsgi application.

7.1.15 Version 1.0.1

Release on Aug 19th, 2013

- Documentation improvement
- Bugfix for server #29
- Bugfix for Task #34

7.1.16 Version 1.0.0

Released on May 9th, 2013

- Redesign the compiler
- Various bugfix

7.1.17 Version 0.11

Released on Nov 7th, 2012

- Redesign server
- remove notification

7.1.18 Version 0.8

Released on Jul 10th, 2012

- Static Server support root page
- Don't compile at first start

7.1.19 Version 0.7

Released on Jun 20th, 2012

- Static Server support index
- Dynamic watch directory changes

7.1.20 Version 0.6

Release on Jun 18th, 2012

- Add static server, 127.0.0.1:35729

7.1.21 Version 0.5

Release on Jun 18th, 2012

- support for python3

7.1.22 Version 0.4

Release on May 8th, 2012

- bugfix for notify (sorry)

7.1.23 Version 0.3

Release on May 6th, 2012

- bugfix for compiler alias
- raise error for CommandCompiler
- add comand-line feature
- get static file from internet

7.1.24 Version 0.2

Release on May 5th, 2012.

- bugfix
- performance improvement
- support for notify-OSD
- alias of compilers

7.1.25 Version 0.1

Released on May 4th, 2012.

CHAPTER 8

Contact

Have any trouble? Want to know more?

- Follow me on GitHub for the latest updates.
- Follow me on Twitter (most tweets are in Chinese).
- Send Email to me.

Python Module Index

l

Index