Django HTTP Proxy

Release 0.4.3-dirty

February 10, 2016

Contents

Django HTTP Proxy provides simple HTTP proxy functionality for the Django web development framework.

CHAPTER 1

Contents

1.1 Introduction

Django HTTP Proxy provides simple HTTP proxy functionality for the Django web development framework. It allows you make requests to an external server by requesting them from the main server running your Django application. In addition, it allows you to record the responses to those requests and play them back at any time.

One possible use for this application (actually, the reason it was developed) is to allow for easy development of Ajax applications against a live server environment:

- Avoid typical cross-domain issues while developing an Ajax application based on live data from another server.
- **Record responses and play them back at a later time:**
    - Use "live" data, even when you are developing offline
    - Speedy responses instead of having to wait for a remote server
- Manually edit recorded responses via the Django admin interface

Combined with the standard Django development server, you have a useful toolkit for developing HTML5/Ajax applications.

Django HTTP Proxy is licensed under an MIT-style permissive license and maintained on Github.

1.2 Installation & configuration

1.2.1 Requirements

Django HTTP Proxy should be compatible with any Python 2.x version from 2.5 and up and a relatively recent version of Django. It is compatible with Django 1.7.

1.2.2 Installation

The easiest way to install the latest version of Django HTTP Proxy is using pip:

```
$ pip install django-http-proxy
```

Alternatively, you can manually download the package from the Python Package Index or from the Github repository.

Next, you need to add "httpproxy" to the `INSTALLED_APPS` list in your Django settings module (typically `settings.py`):

```
INSTALLED_APPS = (
    ...
    'httpproxy',
)
```

Finally, install the database tables:

```
$ python manage.py syncdb
```

Note: If you are only interested in using Django HTTP Proxy as a live proxy and don't care about recording/playing back requests and responses, simply do not add it to your `INSTALLED_APPS` and no database tables will be created.

1.2.3 Configuration

The core of Django HTTP Proxy is a class-based Django view, `httpproxy.views.HttpProxy`.

To use Django HTTP Proxy, you create an entry in your `urls.py` that forwards requests to the `HttpProxy` view class, e.g.:

```
from httpproxy.views import HttpProxy

urlpatterns += patterns('',
    (r'^proxy/(?P<url>.*)$',
        HttpProxy.as_view(base_url='http://www.python.org/')),
)
```

Given the above url config, request matching `/proxy/<any-url>` will be handled by the configured `HttpProxy` view instance and forwarded to `http://www.python.org/<any-url>`.

Note: Older versions of Django HTTP Proxy only supported a single proxy per Django project, which had to be configured using a Django setting:

```
PROXY_BASE_URL = 'http://www.python.org/'
```

Naturally, you can easily replicate this behavior using the new class-based view syntax:

```
from django.conf import settings
from httpproxy.views import HttpProxy

urlpatterns += patterns('',
    (r'^proxy/(?P<url>.*)$',
        HttpProxy.as_view(base_url=settings.PROXY_BASE_URL)),
)
```

1.3 API documentation

Django HTTP Proxy - A simple HTTP proxy for the Django framework.

1.3.1 `httpproxy.views`

1.3.2 `httpproxy.models`

1.3.3 `httpproxy.recorder`

1.3.4 `httpproxy.exceptions`

Some generic exceptions that can occur in Django HTTP Proxy.

exception `httpproxy.exceptions.`**`ResponseUnsupported`**

Raised by `httpproxy.recorder.ProxyRecorder.record()` when it cannot a response (e.g. because it has an unsupported content type).

exception `httpproxy.exceptions.`**`RequestNotRecorded`**

Raised by `httpproxy.recorder.ProxyRecorder.playback()` when a request is made for a URL that has not previously recorded yet.

1.4 Changes

1.4.1 0.4.3 (2016-02-10)

- Merged pull request from Ihor Kucher (thanks!): add support of post requests & update headers setting

1.4.2 0.4.2 (2016-02-10)

- Merged pull request from Don Naegely (thanks!): Generate missing migrations

1.4.3 0.4.1

- Python 3 compatibility (requires Django >= 1.6)
- Fix duplicated forward slashes in urls

1.4.4 0.4

- Migration from Bitbucket (Mercurial) to Github
- Refactored main view using Django class-based views (see `HttpProxy`)
- Removed basic authentication support (*PROXY_USER* and *PROXY_PASSWORD*); may be added back later on.
- Finally merged back Django 1.6 fixes by Petr Dlouhý (thanks!)
- Merged pull request from Garrett Seward (thanks!)
- Added Django 1.7 compatibility
- Added database migrations (Django 1.7 and higher only)
- Updated and improvement the documentation (including *API documentation*)
- Added an `example` project for reference
- Using urllib2 instead of httplib2

- Using setuptools instead of distutils
- Using versioneer2 for package versioning
- Removed some unused imports and did some further code cleanup

1.4.5 0.3.2

- Limited display of request querystring in admin screen to 50 characters

1.4.6 0.3.1

- Fixed 250 character limitation for querystring in Recorded Request (issue #2)
- Added new Request Parameter model; requires `./manage.py reset httpproxy && ./manage.py syncdb`

1.4.7 0.3

- Fixed Python 2.5 support by removing use of `__package__`
- Implemented request path "normalization", fixing record and playback if the proxy is URL-configured anywhere other than directly in the root.
- Added experimental `PROXY_REWRITE_RESPONSES` settings to fix paths to resources (images, javascript, etc) on the same domain if `httpproxy` is not configured at the root.

1.4.8 0.2.2

- Removed print statement I accidentally left behind.

1.4.9 0.2.1

- Fixed issue #1; Unsupported content types are now silently ignored.
- Added `PROXY_IGNORE_UNSUPPORTED` setting to control the behavior for handling unsupported responses.

1.4.10 0.2

- Added recording and playback functionality
- Improved handling of `httpproxy`-specific settings
- Started using Sphinx for documentation

1.4.11 0.1

- Initial release
- Basic HTTP proxy functionality based on a blog post by Will Larson

1.5 Credits

Django HTTP Proxy was created by Yuri van der Meer, inspired by a blog post by Will Larson.

Contributions were made by Petr Dlouhý and Garrett Seward. (thanks!)

1.6 Contributing

Note: This project has recently moved from Bitbucket to Github.

If you have any contributions, feel free to fork Django HTTP Proxy.

1.6.1 Development setup

To set up the project for local development:

```
$ git clone https://github.com/yvandermeer/django-http-proxy.git
$ mkvirtualenv django-http-proxy
$ pip install -r requirements.txt
$ python example/manage.py syncdb
$ python example/manage.py runserver
```

Finally, point your browser to http://127.0.0.1:8000/python/ and you should see something that resembles what you see on http://www.python.org/.

1.6.2 Building the documentation

Documention is provided in Sphinx format in the *docs* subdirectory. To build the HTML version of the documentation yourself:

```
$ cd docs
$ make html
```

1.7 License

Python Module Index

Index