Scrapy Documentation

Release 0.3

Scrapy developers

May 29, 2019

First steps

This documentation contains everything you need to know about Scrapy-Cookies.

CHAPTER 1

First steps

1.1 Scrapy-Cookies at a glance

Scrapy-Cookies is a downloader middleware for Scrapy.

Even though Scrapy-Cookies was originally designed for cookies save and restore (manage the login session), it can also be used to share cookies between various spider nodes.

1.1.1 Walk-through of an example spider

In order to show you what Scrapy-Cookies brings to the table, we'll walk you through an example of a Scrapy project's settings with Scrapy-Cookies using the simplest way to save and restore the cookies.

Here's the code for settings that uses in memory as storage:

```
DOWNLOADER_MIDDLEWARES.update({
    'scrapy.downloadermiddlewares.cookies.CookiesMiddleware': None,
    'scrapy_cookies.downloadermiddlewares.cookies.CookiesMiddleware': 700,
})

COOKIES_ENABLED = True

COOKIES_PERSISTENCE = True
COOKIES_PERSISTENCE_DIR = 'cookies'

# -------------------------------------------------------------------------------
# IN MEMORY STORAGE
# -------------------------------------------------------------------------------

COOKIES_STORAGE = 'scrapy_cookies.storage.in_memory.InMemoryStorage'
```

Put this in your project's settings, and run your spider.

When this finishes you will have a cookies file in the folder .scrapy under your project folder. The file cookies is the pickled object contained cookies from your spider.

What just happened?

When you run your spider, this middleware initializes all objects related to maintaining cookies.

The crawl starts to send requests and receive responses, at the same time this middleware extracts and sets the cookies from and to requests and responses.

When the spider stopped, this middleware will save the cookies to the path defined in COOKIES_PERSISTENCE_DIR.

1.1.2 What else?

You've seen how to save and store cookies with Scrapy-Cookies. And this middleware provides an interface to let you customize your own cookies storage ways, such as:

- In-memory storage, with ultra-fast speed to process
- SQLite storage, with ultra-fast speed when uses memory database, and easy to read and sharing with other process on disk databases
- Other database like MongoDB, MySQL, even HBase to integrate with other programmes across your

1.1.3 What's next?

The next steps for you are to *install Scrapy-Cookies*, *follow through the tutorial* to learn how to create a project with Scrapy-Cookies and join the community. Thanks for your interest!

1.2 Installation guide

1.2.1 Installing Scrapy

Scrapy-Cookies runs on Python 2.7 and Python 3.4 or above under CPython (default Python implementation) and PyPy (starting with PyPy 5.9).

You can install Scrapy-Cookies and its dependencies from PyPI with:

```
pip install Scrapy-Cookies
```

We strongly recommend that you install Scrapy and Scrapy-Cookies in *a dedicated virtualenv*, to avoid conflicting with your system packages.

For more detailed and platform specifics instructions, read on.

Things that are good to know

Scrapy-Cookies is written in pure Python and depends on a few key Python packages (among others):

- Scrapy, of course
- PyMongo

- redis-py
- ujson

The minimal versions which Scrapy-Cookies is tested against are:

- Scrapy 1.5.0

Scrapy-Cookies may work with older versions of these packages but it is not guaranteed it will continue working because it's not being tested against them.

Using a virtual environment (recommended)

TL;DR: We recommend installing Scrapy-Cookies inside a virtual environment on all platforms.

Python packages can be installed either globally (a.k.a system wide), or in user-space. We do not recommend installing Scrapy and Scrapy-Cookies system wide.

Instead, we recommend that you install Scrapy and Scrapy-Cookies within a so-called "virtual environment" (virtualenv). Virtualenvs allow you to not conflict with already-installed Python system packages (which could break some of your system tools and scripts), and still install packages normally with pip (without sudo and the likes).

To get started with virtual environments, see virtualenv installation instructions. To install it globally (having it globally installed actually helps here), it should be a matter of running:

```
$ [sudo] pip install virtualenv
```

Check this user guide on how to create your virtualenv.

Note: If you use Linux or OS X, virtualenvwrapper is a handy tool to create virtualenvs.

Once you have created a virtualenv, you can install Scrapy-Cookies inside it with pip, just like any other Python package. (See *platform-specific guides* below for non-Python dependencies that you may need to install beforehand).

Python virtualenvs can be created to use Python 2 by default, or Python 3 by default.

- If you want to install Scrapy-Cookies with Python 3, install Scrapy-Cookies within a Python 3 virtualenv.
- And if you want to install Scrapy-Cookies with Python 2, install Scrapy-Cookies within a Python 2 virtualenv.

1.2.2 Platform specific installation notes

Windows

Same as Scrapy.

Ubuntu 14.04 or above

Same as Scrapy.

Mac OS X

Same as Scrapy.

PyPy

Same as Scrapy.

1.3 Scrapy-Cookies Tutorial

In this tutorial, we'll assume that Scrapy-Cookies is already installed on your system. If that's not the case, see *Installation guide*.

This tutorial will walk you through these tasks:

1. Use various storage classes in this middleware
2. Save cookies on disk

1.3.1 Use various storage classes in this middleware

Before you start scraping, just put the following code into your settings.py:

```
DOWNLOADER_MIDDLEWARES.update({
    'scrapy.downloadermiddlewares.cookies.CookiesMiddleware': None,
    'scrapy_cookies.downloadermiddlewares.cookies.CookiesMiddleware': 700,
})
```

With the default settings of this middleware, a in-memory storage will be used.

There is a storage named SQLiteStorage. If you want to use it instead of the in-memory one, simple put the following code below the previous one:

```
COOKIES_STORAGE = 'scrapy_cookies.storage.sqlite.SQLiteStorage'
COOKIES_SQLITE_DATABASE = ':memory:'
```

There are other storage classes provided with this middleware, please refer to *Storage*.

When you implement your own storage, you can set `COOKIES_STORAGE` to your own one.

1.3.2 Save cookies and restore in your next run

By default this middleware would not save the cookies. When you need to keep the cookies for further usage, for example a login cookie, you wish to save the cookies on disk for next run.

This middleware provides this ability with one setting:

```
COOKIES_PERSISTENCE = True
```

Most of time the file saved cookies is named `cookies` under the folder `.scrapy`. If you want to change it, use this setting:

```
COOKIES_PERSISTENCE_DIR = 'your-cookies-path'
```

After these settings, this middleware would load the previous saved cookies in the next run.

Note: Please keep the storage is the same class when you want save the cookies and restore them. The cookies persistence file is not compatible between different storage classes.

Note: This feature depends on the storage class used.

1.3.3 Next steps

This tutorial covered only the basics of Scrapy-Cookies, but there's a lot of other features not mentioned here. Check the *What else?* section in *Scrapy-Cookies at a glance* chapter for a quick overview of the most important ones.

You can continue from the section *Basic concepts* to know more about this middleware, storage and other things this tutorial hasn't covered. If you prefer to play with an example project, check the *Examples* section.

1.4 Examples

The best way to learn is with examples, and Scrapy-Cookies is no exception. For this reason, there is an example project with Scrapy-Cookies named grouponbot, that you can use to play and learn more about Scrapy-Cookies. It contains one spiders for https://www.groupon.com.au, only crawl the first page and save the cookies.

The grouponbot project is available at: https://github.com/grammy-jiang/scrapy-enhancement-examples. You can find more information about it in the project's README.

If you're familiar with git, you can checkout the code. Otherwise you can download the project as a zip file by clicking here.

Scrapy-Cookies at a glance Understand what Scrapy-Cookies is and how it can help you.

Installation guide Get Scrapy-Cookies installed on your computer.

Scrapy-Cookies Tutorial Write your first project with Scrapy-Cookies.

Examples Learn more by playing with a pre-made project with Scrapy-Cookies.

CHAPTER 2

Basic concepts

2.1 CookiesMiddleware

This is the downloader middleware to inject cookies into requests and extract cookies from responses.

This middleware mostly inherits the one from Scrapy, which implements the interface of downloader middleware. With minimum changes, now it supports the storage class which implements a certain interface (actually MutableMapping).

2.2 Storage

The class of storage is the one implementing MutableMapping interface. There are some storage classes provided with this middleware:

2.2.1 InMemoryStorage

class `scrapy_cookies.storage.in_memory.`**`InMemoryStorage`**

This storage enables keeping cookies inside the memory, to provide ultra fast read and write cookies performance.

2.2.2 SQLiteStorage

class `scrapy_cookies.storage.sqlite.`**`SQLiteStorage`**

This storage enables keeping cookies in SQLite, which supports already by Python.

The following settings can be used to configure this storage:

- `COOKIES_SQLITE_DATABASE`

2.2.3 MongoStorage

class scrapy_cookies.storage.mongo.**MongoStorage**

This storage enables keeping cookies in MongoDB.

The following settings can be used to configure this storage:

- *COOKIES_MONGO_MONGOCLIENT_HOST*
- *COOKIES_MONGO_MONGOCLIENT_PORT*
- *COOKIES_MONGO_MONGOCLIENT_DOCUMENT_CLASS*
- *COOKIES_MONGO_MONGOCLIENT_TZ_AWARE*
- *COOKIES_MONGO_MONGOCLIENT_CONNECT*
- *COOKIES_MONGO_MONGOCLIENT_KWARGS*
- *COOKIES_MONGO_DATABASE*
- *COOKIES_MONGO_COLLECTION*

2.2.4 RedisStorage

class scrapy_cookies.storage.redis.**RedisStorage**

This storage enables keeping cookies in Redis.

The following settings can be used to configure this storage:

- *COOKIES_REDIS_HOST*
- *COOKIES_REDIS_PORT*
- *COOKIES_REDIS_DB*
- *COOKIES_REDIS_PASSWORD*
- *COOKIES_REDIS_SOCKET_TIMEOUT*
- *COOKIES_REDIS_SOCKET_CONNECT_TIMEOUT*
- *COOKIES_REDIS_SOCKET_KEEPALIVE*
- *COOKIES_REDIS_SOCKET_KEEPALIVE_OPTIONS*
- *COOKIES_REDIS_CONNECTION_POOL*
- *COOKIES_REDIS_UNIX_SOCKET_PATH*
- *COOKIES_REDIS_ENCODING*
- *COOKIES_REDIS_ENCODING_ERRORS*
- *COOKIES_REDIS_CHARSET*
- *COOKIES_REDIS_ERRORS*
- *COOKIES_REDIS_DECODE_RESPONSES*
- *COOKIES_REDIS_RETRY_ON_TIMEOUT*
- *COOKIES_REDIS_SSL*
- *COOKIES_REDIS_SSL_KEYFILE*
- *COOKIES_REDIS_SSL_CERTFILE*

- *COOKIES_REDIS_SSL_CERT_REQS*
- *COOKIES_REDIS_SSL_CA_CERTS*
- *COOKIES_REDIS_MAX_CONNECTIONS*

2.3 Settings

The default settings of this middleware keeps the same behaviour as the one in Scrapy.

As an enhancement, there are some settings added in this middleware:

2.3.1 COOKIES_PERSISTENCE

Default: `False`

Whether to enable this cookies middleware save the cookies on disk. If disabled, no cookies will be saved on disk.

Notice that this setting only affects when the storage uses memory as cookies container.

2.3.2 COOKIES_PERSISTENCE_DIR

Default: `cookies`

When `COOKIES_PERSISTENCE` is True, the storage which use memory as cookies container will save the cookies in the file `cookies` under the folder `.scrapy` in your project, while if the storage does not use memory as cookies container will not affect by this setting.

2.3.3 COOKIES_STORAGE

Default: `scrapy_cookies.storage.in_memory.InMemoryStorage`

With this setting, the storage can be specified. There are some storage classes provided with this middleware by default:

- *scrapy_cookies.storage.in_memory.InMemoryStorage*
- *scrapy_cookies.storage.sqlite.SQLiteStorage*
- *scrapy_cookies.storage.mongo.MongoStorage*

2.3.4 COOKIES_MONGO_MONGOCLIENT_HOST

Default: `localhost`

Hostname or IP address or Unix domain socket path of a single mongod or mongos instance to connect to, or a mongodb URI, or a list of hostnames / mongodb URIs. If host is an IPv6 literal it must be enclosed in '[' and ']' characters following the RFC2732 URL syntax (e.g. '[::1]' for localhost). Multihomed and round robin DNS addresses are not supported.

Please refer to mongo_client.

2.3.5 COOKIES_MONGO_MONGOCLIENT_PORT

Default: `27017`

Port number on which to connect.

Please refer to mongo_client.

2.3.6 COOKIES_MONGO_MONGOCLIENT_DOCUMENT_CLASS

Default: `dict`

Default class to use for documents returned from queries on this client.

Please refer to mongo_client.

2.3.7 COOKIES_MONGO_MONGOCLIENT_TZ_AWARE

Default: `False`

If True, datetime instances returned as values in a document by this MongoClient will be timezone aware (otherwise they will be naive).

Please refer to mongo_client.

2.3.8 COOKIES_MONGO_MONGOCLIENT_CONNECT

Default: `True`

If True (the default), immediately begin connecting to MongoDB in the background. Otherwise connect on the first operation.

Please refer to mongo_client.

2.3.9 COOKIES_MONGO_MONGOCLIENT_KWARGS

Please refer to mongo_client.

2.3.10 COOKIES_MONGO_DATABASE

Default: `cookies`

The name of the database - a string. If None (the default) the database named in the MongoDB connection URI is returned.

Please refer to get_database.

2.3.11 COOKIES_MONGO_COLLECTION

Default: `cookies`

The name of the collection - a string.

Please refer to get_collection.

2.3.12 COOKIES_REDIS_HOST

Please refer to redis-py's documentation.

2.3.13 COOKIES_REDIS_PORT

Please refer to redis-py's documentation.

2.3.14 COOKIES_REDIS_DB

Please refer to redis-py's documentation.

2.3.15 COOKIES_REDIS_PASSWORD

Please refer to redis-py's documentation.

2.3.16 COOKIES_REDIS_SOCKET_TIMEOUT

Please refer to redis-py's documentation.

2.3.17 COOKIES_REDIS_SOCKET_CONNECT_TIMEOUT

Please refer to redis-py's documentation.

2.3.18 COOKIES_REDIS_SOCKET_KEEPALIVE

Please refer to redis-py's documentation.

2.3.19 COOKIES_REDIS_SOCKET_KEEPALIVE_OPTIONS

Please refer to redis-py's documentation.

2.3.20 COOKIES_REDIS_CONNECTION_POOL

Please refer to redis-py's documentation.

2.3.21 COOKIES_REDIS_UNIX_SOCKET_PATH

Please refer to redis-py's documentation.

2.3.22 COOKIES_REDIS_ENCODING

Please refer to redis-py's documentation.

2.3.23 COOKIES_REDIS_ENCODING_ERRORS

Please refer to redis-py's documentation.

2.3.24 COOKIES_REDIS_CHARSET

Please refer to redis-py's documentation.

2.3.25 COOKIES_REDIS_ERRORS

Please refer to redis-py's documentation.

2.3.26 COOKIES_REDIS_DECODE_RESPONSES

Please refer to redis-py's documentation.

2.3.27 COOKIES_REDIS_RETRY_ON_TIMEOUT

Please refer to redis-py's documentation.

2.3.28 COOKIES_REDIS_SSL

Please refer to redis-py's documentation.

2.3.29 COOKIES_REDIS_SSL_KEYFILE

Please refer to redis-py's documentation.

2.3.30 COOKIES_REDIS_SSL_CERTFILE

Please refer to redis-py's documentation.

2.3.31 COOKIES_REDIS_SSL_CERT_REQS

Please refer to redis-py's documentation.

2.3.32 COOKIES_REDIS_SSL_CA_CERTS

Please refer to redis-py's documentation.

2.3.33 COOKIES_REDIS_MAX_CONNECTIONS

Please refer to redis-py's documentation.

CookiesMiddleware Extract cookies from response and Restore cookies to request.

Storage Save ,restore and share the cookies.

Settings Learn how to configure Scrapy-Cookies and see all available settings.

CHAPTER 3

Extending Scrapy-Cookies

Storage Customize how the storage save, restore and share the cookies

Python Module Index

Index