BrowserMob Proxy Documentation

Release 0.6.0

David Burns

May 23, 2018

Contents

Python client for the BrowserMob Proxy 2.0 REST API.

CHAPTER 1

How to install

BrowserMob Proxy is available on PyPI, so you can install it with `pip`:

```
$ pip install browsermob-proxy
```

Or with *easy_install*:

```
$ easy_install browsermob-proxy
```

Or by cloning the repo from GitHub:

```
$ git clone git://github.com/AutomatedTester/browsermob-proxy-py.git
```

Then install it by running:

```
$ python setup.py install
```

CHAPTER 2

How to use with selenium-webdriver

Manually:

```
from browsermobproxy import Server
server = Server("path/to/browsermob-proxy")
server.start()
proxy = server.create_proxy()

from selenium import webdriver
profile  = webdriver.FirefoxProfile()
profile.set_proxy(proxy.selenium_proxy())
driver = webdriver.Firefox(firefox_profile=profile)


proxy.new_har("google")
driver.get("http://www.google.co.uk")
proxy.har # returns a HAR JSON blob

server.stop()
driver.quit()
```

CHAPTER 3

How to Contribute

3.1 Getting Started

- Fork the repository on GitHub - well... duh :P
- Create a virtualenv: *virtualenv venv*
- Activate the virtualenv: . *venv/bin/activate*
- Install the package in develop mode: *python setup.py develop*
- Install requirements: *pip install -r requirements.txt*
- Run the tests to check that everything was successful: ‘py.test tests

3.2 Making Changes

- Create a topic branch from where you want to base your work. * This is usually the master branch. * Only target release branches if you are certain your fix must be on that

  branch.

  - To quickly create a topic branch based on master; *git checkout -b /my_contribution master*. Please avoid working directly on the *master* branch.
- Make commits of logical units.
- Check for unnecessary whitespace with *git diff –check* before committing.
- Make sure you have added the necessary tests for your changes.
- Run _all_ the tests to assure nothing else was accidentally broken.

3.3 Submitting Changes

- Push your changes to a topic branch in your fork of the repository.
- Submit a pull request to the main repository
- After feedback has been given we expect responses within two weeks. After two weeks will may close the pull request if it isn't showing any activity

Contents:

3.3.1 `client` Package

class browsermobproxy.**Client**(*url*, *params=None*, *options=None*)
Initialises a new Client object

Parameters

- **url** – This is where the BrowserMob Proxy lives
- **params** – URL query (for example httpProxy and httpsProxy vars)
- **options** – Dictionary that can contain the port of an existing proxy to use (for example 'existing_proxy_port_to_use')

add_to_capabilities(*capabilities*)
Adds an 'proxy' entry to a desired capabilities dictionary with the BrowserMob proxy information

Parameters **capabilities** – The Desired capabilities object from Selenium WebDriver

basic_authentication(*domain*, *username*, *password*)
This add automatic basic authentication

Parameters

- **domain** (*str*) – domain to set authentication credentials for
- **username** (*str*) – valid username to use when authenticating
- **password** (*str*) – valid password to use when authenticating

blacklist(*regexp*, *status_code*)
Sets a list of URL patterns to blacklist

Parameters

- **regex** (*str*) – a comma separated list of regular expressions
- **status_code** (*int*) – the HTTP status code to return for URLs that do not match the blacklist

clear_all_rewrite_url_rules()
Clears all URL rewrite rules :return: status code

clear_dns_cache()
Clears the DNS cache associated with the proxy instance

close()
shuts down the proxy and closes the port

har
Gets the HAR that has been recorded

headers (*headers*)
This sets the headers that will set by the proxy on all requests

Parameters **headers** (*dict*) – this is a dictionary of the headers to be set

limits (*options*)
Limit the bandwidth through the proxy.

Parameters **options** (*dict*) – A dictionary with all the details you want to set. downstream_kbps - Sets the downstream kbps upstream_kbps - Sets the upstream kbps latency - Add the given latency to each HTTP request

new_har (*ref=None*, *options=None*, *title=None*)
This sets a new HAR to be recorded

Parameters

- **ref** (*str*) – A reference for the HAR. Defaults to None
- **options** (*dict*) – A dictionary that will be passed to BrowserMob Proxy with specific keywords. Keywords are:
  - captureHeaders: Boolean, capture headers
  - captureContent: Boolean, capture content bodies
  - captureBinaryContent: Boolean, capture binary content
- **title** (*str*) – the title of first har page. Defaults to ref.

new_page (*ref=None*, *title=None*)
This sets a new page to be recorded

Parameters

- **ref** (*str*) – A reference for the new page. Defaults to None
- **title** (*str*) – the title of new har page. Defaults to ref.

proxy_ports
Return a list of proxy ports available

remap_hosts (*address=None*, *ip_address=None*, *hostmap=None*)
Remap the hosts for a specific URL

Parameters

- **address** (*str*) – url that you wish to remap
- **ip_address** (*str*) – IP Address that will handle all traffic for the address passed in
- ****hostmap** – Other hosts to be added as keyword arguments

request_interceptor (*js*)
Executes the java/js code against each response HttpRequest request, HttpMessageContents contents, HttpMessageInfo messageInfo are available objects to interact with. :param str js: the js/java code to execute

response_interceptor (*js*)
Executes the java/js code against each response HttpRequest request, HttpMessageContents contents, HttpMessageInfo messageInfo are available objects to interact with. :param str js: the js/java code to execute

retry (*retry_count*)
Retries. No idea what its used for, but its in the API...

Parameters **retry_count** (*int*) – the number of retries

rewrite_url(*match*, *replace*)
Rewrites the requested url.

Parameters

- **match** – a regex to match requests with
- **replace** – unicode a string to replace the matches with

selenium_proxy()
Returns a Selenium WebDriver Proxy class with details of the HTTP Proxy

timeouts(*options*)
Configure various timeouts in the proxy

Parameters **options** (*dict*) – A dictionary with all the details you want to set. request - request timeout (in seconds) read - read timeout (in seconds) connection - connection timeout (in seconds) dns - dns lookup timeout (in seconds)

wait_for_traffic_to_stop(*quiet_period*, *timeout*)
Waits for the network to be quiet

Parameters

- **quiet_period** (*int*) – number of milliseconds the network needs to be quiet for
- **timeout** (*int*) – max number of milliseconds to wait

webdriver_proxy()
Returns a Selenium WebDriver Proxy class with details of the HTTP Proxy

whitelist(*regexp*, *status_code*)
Sets a list of URL patterns to whitelist

Parameters

- **regex** (*str*) – a comma separated list of regular expressions
- **status_code** (*int*) – the HTTP status code to return for URLs that do not match the whitelist

3.3.2 server Package

class browsermobproxy.**Server**(*path='browsermob-proxy'*, *options=None*)
Initialises a Server object

Parameters

- **path** (*str*) – Path to the browsermob proxy batch file
- **options** (*dict*) – Dictionary that can hold the port. More items will be added in the future. This defaults to an empty dictionary

start(*options=None*)
This will start the browsermob proxy and then wait until it can interact with it

Parameters **options** (*dict*) – Dictionary that can hold the path and filename of the log file with resp. keys of *log_path* and *log_file*

stop()
This will stop the process running the proxy

3.3.3 Indices and tables

- genindex
- modindex
- search

Python Module Index

b

Index