httpsrvvcr Documentation

Release 0.1.7

2GIS

Aug 08, 2016

Contents

VCR recorder for httpsrv API mocking library. Works as a proxy recording real API calls to yaml "vcr tape" that can further be used as httpsrv fixture

class recorder.**ProxyHandler**(*application*, *request*, ***kwargs*)

Implementation of a tornado.web.RequestHandler that proxies any recieved request to a target URL and recorders everything that passes through into a given writer

initialize(*httpclient*, *target*, *writer*)

Initializes a handler, overrides standard tornado.web.RequestHandler method

Parameters

- **httpclient** (*tornado.httpclient.AsyncHTTPClient*) – httpclient that will be used to make requests to target URL
- **target** (*str*) – target API URL to proxy requests to
- **writer** (VcrWriter) – vcr writer that will be used to output recorded requests

class recorder.**VcrWriter**(*writer*, *json*, *no_headers=False*, *skip_methods=None*)

Converts tornado.httputil.HTTPServerRequest and tornado.httpclient.HTTPResponse objects into a vcr output utilizing an underlying writer

Parameters

- **writer** (*object*) – writer object that supports write(data) interface
- **json** (*json*) – json module from standard library
- **no_headers** (*bool*) – if True then no headers will be recorded for request or resposne

write(*request*, *response*)

Writes a vcr output in a form of a dict from given tornado.httputil.HTTPServerRequest and tornado.httpclient.HTTPResponse

Parameters

- **request** (*tornado.httputil.HTTPServerRequest*) – server request
- **response** (*tornado.httpclient.HTTPResponse*) – client response

class recorder.**YamlWriter**(*writer*, *yaml*)

Acts as a decorator for the wrapped writer object. Any data given to *YamlWriter.write()* will be converted to yaml string and passde to the underlying writer

Parameters

- **writer** (*object*) – writer object that will recieve yaml string. Must support write(str)
- **yaml** (*yaml*) – yaml encoder, must support pyyaml-like interface

write(*data*)

Writes given data as a yaml string to an underlying stream

Parameters **data** (*any*) – object that will be conveted to yaml string

recorder.**run**(*port*, *target*, *no_headers=False*, *skip_methods=None*)

Starts a vcr proxy on a given port using target as a request destination

Parameters

- **port** (*int*) – port the proxy will bind to
- **target** (*str*) – URL to proxy requests to, must be passed with protocol, e.g. http://some-url.com

- **no_headers** (*bool*) – if `True` then no headers will be recorded for request or resposne
- **skip_methods** (*list*) – recorder will not write any requests with provided methods to output

`recorder.`**`stop`**`()`
Stops currently running vcr proxy

CHAPTER 1

Player

VCR Player plays tapes recorded with `httpsrvvcr.recorder` using `httpsrv.Server` provided

class `player.`**`Player`**(*server*, *add_cors=False*)

Player wraps the `httpsrv.Server` and plays perviously recorded vcr tapes on it

Parameters

- **`server`** (*`httpsrv.Server`*) – server thta will be loaded with rules from vcr tapes
- **`add_cors`** (*`bool`*) – if `True` player will add CORS header to all responses

`load`(*tape_file_name*)

Decorator that can be used on test functions to read vcr tape from file and load current player with it:

```
@player.load('path/to/tape.yaml')
def test_should_do_some_vcr(self):
    pass
```

Parameters **`tape_file_name`** (*`str`*) – tape filename to load

`play`(*tape*)

Loads the server with rules created from tape passed

Parameters **`tape`** (*`dict`*) – vcr tape previously recorded with `httpsrvvcr.recorder`

`player.`**`tape_from_yaml`**(*yaml_text*)

Parses yaml tape into python dictionary

Parameters **`yaml_text`** (*`str`*) – yaml string to parse

CHAPTER 2

Httpsrv VCR

Library for recording http requests into yaml format that can be further understood by httpsrv as a server fixture

2.1 Installation

Package can be obtained from PyPi

```
pip install httpsrvvcr
```

2.2 Usage

Basic usage looks like following:

```
python -m httpsrvvcr.recorder 8080 http://some-api-url.com/api > tape.yaml
```

It is possible to skip headers recording with `--no-headers` flag:

```
python -m httpsrvvcr.recorder 8080 http://some-api-url.com/api --no-headers > tape.yaml
```

Once can also exclude some request methods from output completely:

```
python -m httpsrvvcr.recorder 8080 http://some-api-url.com/api --skip-methods OPTIONS TRACE > tape.ya
```

After vcr tape is recorded one can use `httpsrvvcr.player` module:

```
import unittest

from httpsrv import Server
from httpsrvvcr.player import Player

server = Server(8080).start()
player = Player(server)

class MyTestCase(unittest.TestCase):
    def setUp(self):
        server.reset()

    @player.load('path/to/tape.yaml')
    def test_should_do_something(self):
        pass
```

2.3 Documentation

http://httpsrvvcr.readthedocs.io

p

r

I

L

P

R

S

T

V

W

Y