
expectly Documentation

Release 0.1.8

Hunter Senft-Grupp

Nov 26, 2017

Contents:

1	Installing	3
2	API Examples	5
2.1	Built In Assertion methods	5
2.2	Extending expectly.expect with additional assertions	5
2.3	JSON handling in expectly	6
3	Development	7
4	API	9
5	Indices and tables	11
	Python Module Index	13

Use Cases:

- Making assertions about api responses more readable
- Extending expectly with new assertions to reduce boilerplate

Danger: This library is in alpha. I will try my utmost to not make backwards incompatible changes but it is possible that

CHAPTER 1

Installing

```
$ pip install exactly
```


2.1 Built In Assertion methods

```
# status_code (with status code number)
expect(response).to.have.status_code(200)

# header (asserts the header exists). Sets the header value to be further_
↳tested against
expect(response).to.have.header('Content-Type')

# equals
expect(response).to.have.header('Content-Type').equal('application/json')

# like (does a regex match against the value)
expect(response).to.have.header('Content-Type').like('application')

# exactly_like (does a regex match against the whole value
↳[\w]*$')
expect(response).to.have.header('Content-Type').exactly_like('^application/
↳[\w]*$')

# encoding
expect(response).to.have.encoding('utf-8')

# ok (uses requests.response.ok attribute)
expect(response).to.be.ok
```

2.2 Extending expectly.expect with additional assertions

TODO

2.3 JSON handling in expectly

jsonschema is used for evaluating whether json responses match the relevant schema jmespath is used for navigating/selecting information to test inside of the json response

CHAPTER 3

Development

```
$ git clone https://github.com/huntcsg/expectly.git
$ cd expectly
$ ./utils/manage clean
$ ./utils/manage test
$ ./utils/manage docs
```

1. All pull requests must pass the travis-ci builds
2. All pull requests should include inline (docstring) documentation, updates to built documentation if applicable, and test coverage. This project aspires to be a 100% test coverage library.

class `expectly._expect.expect` (*response_under_test*)

The main class in this library. Provides the test methods.

classmethod `add_assertion` (*name*, *assertion_callable*, *is_property=False*, *predicate_str=None*)

Helper function to add an assertion to the expect class

Parameters

- **name** –
- **assertion_callable** –
- **is_property** –
- **predicate_str** –

Returns

classmethod `add_join_word` (*word*, *predicate_str=None*)

Adds a non semantically meaningful join word, e.g. “to”, “and”, etc

Parameters

- **word** –
- **predicate_str** –

Returns

and_

Returns

assertion

Composes the current human readable assertion

Returns a `str`

be

Returns

contains (*value*)

encoding (*encoding*)

equal (*value*)

equals (*value*)

exactly_like (*regex*)

get ()

A lazy getter for the value under test

have

Returns

header (*header*)

Tests that the header exists. Sets the header to the `value_under_test`

headers (*headers*)

json

like (*regex*)

not_

Changes the test method to be a negative assertion

Returns

null

ok

path (*path*)

schema (*schema*)

status_code (*code*)

test (*value=True*)

The default test method. It asserts the passed value is true with the assertion clause

that

Returns

the

Returns

to

Returns

update_test_method (*fn*)

Given a function calls that function when `expectly.expect.test()` is called.

Parameters **fn** –

Returns

valid

value

CHAPTER 5

Indices and tables

- `genindex`
- `modindex`
- `search`

e

`expectly`, 9

`expectly._expect`, 9

E

`expectly (module)`, 9

`expectly._expect (module)`, 9