noseapp_requests Documentation

Release 0.0.1

Artem Gorokhov <brannt@yandex.ru>

March 26, 2015

Contents

1 Basics **3**
- 1.1 Setting up the extension . . . 3
- 1.2 Making requests . . . 3
- 1.3 Authentication . . . 4

2 Reference **5**
- 2.1 noseapp_requests.base module . . . 5
- 2.2 noseapp_requests.config module . . . 6
- 2.3 noseapp_requests.urlbuilder module . . . 7

3 Indices and tables **9**

Python Module Index **11**

noseapp_requests is intended for use with noseapp library. It provides an extension for working with HTTP requests and REST APIs, based on the requests library.

Contents:

CHAPTER 1

Basics

1.1 Setting up the extension

```
from noseapp.ext.requests import RequestsEx, make_config

endpoint = make_config()
endpoint.configure(
    base_url='http://httpbin.org/',
    key='httpbin'
)
endpoint.session_configure(
    always_return_json=True,
    raise_on_http_error=True
)
requests_ex = RequestsEx(endpoint)
```

1.2 Making requests

Get an API session:

```
>>> api = requests_ex.get_endpoint_session('httpbin', auth=('user', 'pass'))
```

Simple *GET* request (if `always_return_json` option is set, response is automatically parsed as JSON into a `noseapp.datastructures.ModifyDict` which allows access to values via dot):

```
>>> api.get('basic-auth/user/pass')
{u'authenticated': True, u'user': u'user'}
>>> r = api.get('basic-auth/user/pass')
>>> r.authenticated
True
>>> r.user
u'user'
```

GET with query-string parameters:

```
>>> api.get('get', key1='val1')
{u'args': {u'key1': u'val1'},
 <...>
 u'url': u'http://httpbin.org/get?key1=val1'}
```

POST form-encoded data:

```
>>> api.post('post', key1='val1')
{u'form': {u'key1': u'val1'},
 <...>
 u'url': u'http://httpbin.org/post'}
```

POST JSON data:

```
>>> api.post('post', {'key1': 'val1'})
{u'data': u'{"key1": "val1"}',
 u'json': {u'key1': u'val1'},
 <...>
 u'url': u'http://httpbin.org/post'}
```

If `raise_on_http_error` option is set, an exception is raised if response status code is 4xx or 5xx

```
>>> api.get('status/400')
HTTPError: 400 Client Error: BAD REQUEST
```

1.3 Authentication

You can provide custom authentication class for an endpoint via `auth_cls` config parameter:

```
>>> from requests_oauthlib import OAuth2
>>> endpoint.configure(auth_cls=OAuth2)
```

Then you can pass authentication parameter in runtime when getting an API session:

```
>>> api = requests_ex.get_endpoint_session('oauth2', auth=(client_id, client, token))
```

CHAPTER 2

Reference

2.1 noseapp_requests.base module

class `noseapp_requests.base.`**`Endpoint`**(*config*)

Represents an API endpoint.

Parameters **config** (*Config*) – endpoint config

`get_session`(***kwargs*)

Get API session for the endpoint.

If `auth` argument is provided and `auth_cls` parameter was present in config, `auth` will be passed as arguments to `auth_cls` instance. All other arguments will be passed to `Session` as is.

class `noseapp_requests.base.`**`RequestsEx`**(**configs*)

Extension initializer.

Usage:

```
>>> requests_ex = RequestsEx(ENDPOINT1_CONFIG, ENDPOINT2_CONFIG, ...)
```

Each endpoint config represents an API endpoint (basically, a base URL) that you need access to in tests. So, for example, if you need to call your app's API at http://yourapp.test/api/v1/ and also get data from some external source at http://databank.test:6847/, you'll provide two configs to the extension::

```
app_config = make_config()
app_config.configure(base_url='http://yourapp.test/api/v1/')
databank_config = make_config()
databank_config.configure(base_url='http://databank.test:6847/')
requests_ex = RequestsEx(app_config, databank_config)
```

Each config can be provided a `key` that will be used to refer to it. By default, `base_url` is used as key.

`get_endpoint_session`(*endpoint_key*, ***kwargs*)

Get API session for endpoint `endpoint_key`. Any additional `kwargs` will be passed to `Endpoint.get_session()`.

`name` = **'requests'**

exception `noseapp_requests.base.`**`RequestsExError`**

class `noseapp_requests.base.`**`Session`**(*base_url=None*, *url_builder=None*, *always_return_json=False*, *raise_on_http_error=False*, ***session_kwargs*)

Represents an API/requests session. Actual interface is dependent on the `url_builder` provided; by default it mimics `requests` interface.

Parameters

- **base_url** – Base URL for endpoint; if provided, default URL builder will prepend it to all URLs provided in method calls.
- **url_builder** – A custom URL builder callable. See `noseapp_requests.urlbuilder` documentation for details.
- **always_return_json** (*bool*) – Parse all responses to JSON if set.
- **raise_on_http_error** (*bool*) – Raise HTTPError if response status code >= 400.

All other keyword arguments are set directly on `requests.Session`.

Default interface:

get(*url*, ***params*)

post(*url*, *json_object=None*, ***params*)

put(*url*, *json_object=None*, ***params*)

delete(*url*, ***params*)

All keyword parameters for `get` go into query string, for other methods they are urlencoded into request body.

2.2 noseapp_requests.config module

class `noseapp_requests.config.`**Config**

Extension configuration.

configure(***params*)

Set endpoint configuration params.

Parameters

- **key** – A key that will be used to refer to the endpoint.
- **base_url** – Base URL for endpoint.
- **auth_cls** – Authentication class to use with endpoint, e.g. `requests_oauthlib.OAuth2`.

session_configure(***params*)

Set common session params.

Parameters

- **always_return_json** (*bool*) – Parse all responses to JSON if set.
- **raise_on_http_error** (*bool*) – Raise HTTPError if response status code >= 400.

Param

- Any params for `requests.request()` or `requests.Session`

`noseapp_requests.config.`**make_config**()

Return `Config` instance with default values.

2.3 noseapp_requests.urlbuilder module

Here be dragons.

class `noseapp_requests.urlbuilder.`**`BaseUrlBuilder`**(*base_url*)

`SUPPORTED_METHODS` = **('get', 'post', 'put', 'delete')**

CHAPTER 3

Indices and tables

- *genindex*
- *modindex*
- *search*

n

noseapp_requests, 3
noseapp_requests.base, 5
noseapp_requests.config, 6
noseapp_requests.urlbuilder, 7

B

BaseUrlBuilder (class in noseapp_requests.urlbuilder), 7

C

Config (class in noseapp_requests.config), 6
configure() (noseapp_requests.config.Config method), 6

D

delete() (noseapp_requests.base.Session method), 6

E

Endpoint (class in noseapp_requests.base), 5

G

get() (noseapp_requests.base.Session method), 6
get_endpoint_session() (noseapp_requests.base.RequestsEx method), 5
get_session() (noseapp_requests.base.Endpoint method), 5

M

make_config() (in module noseapp_requests.config), 6

N

name (noseapp_requests.base.RequestsEx attribute), 5
noseapp_requests (module), 3
noseapp_requests.base (module), 5
noseapp_requests.config (module), 6
noseapp_requests.urlbuilder (module), 7

P

post() (noseapp_requests.base.Session method), 6
put() (noseapp_requests.base.Session method), 6

R

RequestsEx (class in noseapp_requests.base), 5
RequestsExError, 5

S

Session (class in noseapp_requests.base), 5
session_configure() (noseapp_requests.config.Config method), 6
SUPPORTED_METHODS (noseapp_requests.urlbuilder.BaseUrlBuilder attribute), 7