tornado$_{j}sonapiDocumentation$

Release 0.1.3

Andrew Kravchuk

April 07, 2016

Contents

1 Library Reference **3**
1.1 Handlers 3
1.2 Resources 4
1.3 Exceptions 4

2 Indices and tables **5**

Python Module Index **7**

Contents:

CHAPTER 1

Library Reference

1.1 Handlers

class tornado_jsonapi.handlers.**APIHandler**(*application*, *request*, ***kwargs*)

Basic tornado.web.RequestHandler for JSON API. It handles validating both input and output documents, rendering resources to JSON, error processing and other aspects of JSON API. Subclass it to add extra functionality, e.g. authorization or JSON API extension support.

acceptable(*extensions*)

Return whether server supports given extensions. By default do not support any extensions.

Parameters **sender** (*dict*) – dictionary of Accept header parameters, e.g. {'supported-ext': 'bulk,jsonpatch'}

delete(*id_*)

DELETE method, see spec. Decorate with tornado.gen.coroutine() when subclassing.

get(*id_=None*)

GET method, see spec. Decorate with tornado.gen.coroutine() when subclassing.

patch(*id_*)

PATCH method, see spec. Decorate with tornado.gen.coroutine() when subclassing.

post(*id_=None*)

POST method, see spec. Decorate with tornado.gen.coroutine() when subclassing.

render_resource(*resource*, *nullable=True*)

Utility function

class tornado_jsonapi.handlers.**NotFoundErrorAPIHandler**(*application*, *request*, ***kwargs*)

Handler for 404 error providing correct API response. Do not use it directly, use *not_found_handling_settings()* instead.

tornado_jsonapi.handlers.**not_found_handling_settings**()

Settings dict for tornado.web.Application to use *NotFoundErrorAPIHandler* as default 404 error handler. Use this as follows:

```
application = tornado.web.Application([
    (
        # ... handlers ...
    ),
], **tornado_jsonapi.handlers.not_found_handling_settings())
application.listen()
```

1.2 Resources

1.3 Exceptions

exception tornado_jsonapi.exceptions.**APIError**(*status_code=500*, *details='Unspecified API error'*, **args*, ***kwargs*)

Basic API error for using in client code. Each *APIError* is assigned an unique error ID, which is both written to log and returned to API client (as `id` field of `error` object) to simplify debugging.

Usage example:

```
class MyResource(tornado_jsonapi.resource.Base):
    def read(self, id_):
        if not self.ham():
            raise tornado_jsonapi.exceptions.APIError(details='No ham left!')
```

Parameters

- **status_code** (*int*) – HTTP status code
- **details** (*str*) – Details of the error to be shown to API client and written to log. May contain `%s`-style placeholders, which will be filled in with remaining positional parameters.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

t

tornado_jsonapi, 3
tornado_jsonapi.exceptions, 4
tornado_jsonapi.handlers, 3
tornado_jsonapi.resource, 4

A

acceptable() (tornado_jsonapi.handlers.APIHandler method), 3
APIError, 4
APIHandler (class in tornado_jsonapi.handlers), 3

D

delete() (tornado_jsonapi.handlers.APIHandler method), 3

G

get() (tornado_jsonapi.handlers.APIHandler method), 3

N

not_found_handling_settings() (in module tornado_jsonapi.handlers), 3
NotFoundErrorAPIHandler (class in tornado_jsonapi.handlers), 3

P

patch() (tornado_jsonapi.handlers.APIHandler method), 3
post() (tornado_jsonapi.handlers.APIHandler method), 3

R

render_resource() (tornado_jsonapi.handlers.APIHandler method), 3

T

tornado_jsonapi (module), 3
tornado_jsonapi.exceptions (module), 4
tornado_jsonapi.handlers (module), 3
tornado_jsonapi.resource (module), 4