sprockets.handlers.status

Release 0.1.2

June 09, 2015

Contents

A small handler for reporting application status

CHAPTER 1

Installation

sprockets.handlers.status is available on the Python Package Index and can be installed via pip or easy_install:

```
pip install sprockets.handlers.status
```

CHAPTER 2

API Documentation

sprockets.handlers.status

A small handler for reporting application status

class sprockets.handlers.status.**StatusHandler**(*application*, *request*, ***kwargs*)

Implement a status handler endpoint that can be used to get information about the current service

get(**args*, ***kwargs*)

Tornado RequestHandler GET request endpoint for reporting status

Parameters

- **args** (*list*) – positional args
- **kwargs** (*dict*) – keyword args

sprockets.handlers.status.**set_application**(*name*)

Set the application name that is reported in the status.

Parameters **name** (*str*) – The application name

2.1 Examples

The following example demonstrates how to initialize the status handler for the base application status page.

```
import tornado.ioloop
import tornado.web
from sprockets.handlers import status


application = tornado.web.Application([
    ('/status', status.StatusHandler),
])

if __name__ == '__main__':

    # Set the application name to the local package
    status.set_application('mypackage')

    application.listen(8888)
    tornado.ioloop.IOLoop.current().start()
```

You can change the status by setting the `status` attribute on `RequestHandler.application`, but note this does it for the single running process and should be coordinated across all backends using an external synchronization mechanism.

The following example uses Consul to determine if a request is in maintenance mode.

```
import json
import socket

from tornado import gen
from tornado import httpclient
from sprockets.handlers import status

class ConsulStatusHandler(status.StatusHandler):

    HEALTH_URL_FORMAT = 'http://localhost:8500/v1/health/node/{0}

    @gen.coroutine
    def prepare(self):
        result = yield self._maintenance_enabled()
        if result:
            setattr(self.application, 'status', status.MAINTENANCE)
        else:
            setattr(self.application, 'status', status.OK)

    @gen.coroutine
    def _maintenance_enabled(self)
        client = httpclient.AsyncHTTPClient()
        url = self.HEALTH_URL_FORMAT.format(socket.gethostname())
        result = yield client.fetch()
        return self._in_maintenance(json.loads(result.body))

    @staticmethod
    def _in_maintenance(self, checks):
        for check in checks:
             if check['CheckID'] == '_node_maintenance':
                return check['Status'] == 'critical'
        return False
```

CHAPTER 3

Version History

See `history`

CHAPTER 4

Issues

Please report any issues to the Github project at https://github.com/sprockets/sprockets.handlers.status/issues

CHAPTER 5

Source

`sprockets.handlers.status` source is available on Github at https://github.com/sprockets/sprockets.handlers.status

CHAPTER 6

License

`sprockets.handlers.status` is released under the 3-Clause BSD license.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

S

G

S