Django-HTTPolice Documentation

Vasiliy Faronov

Jun 27, 2019

Contents

1 Example 3

2 Installation 5

3 Viewing the backlog 7

4 Raising on notices 9

5 Silencing unwanted notices 11

Django-HTTPolice is a package that integrates HTTPolice into Django 1.11+.

For recent changes in Django-HTTPolice, see the changelog.

CHAPTER 1

Example

For a small example of Django-HTTPolice in action, see the example directory in the Git repo.

CHAPTER 2

Installation

```
$ pip3 install Django-HTTPolice
```

This package provides `django_httpolice.HTTPoliceMiddleware`. Add it to your `MIDDLEWARE` list, as close to the top as possible:

```
MIDDLEWARE = [
    'django_httpolice.HTTPoliceMiddleware',
    'django.middleware.common.CommonMiddleware',
    # ...
]
```

This middleware does **nothing** until you also set the `HTTPOLICE_ENABLE` setting to `True`.

When enabled, the middleware checks all exchanges passing through it. Then, there are two different ways to see the results of these checks.

CHAPTER 3

Viewing the backlog

All exchanges checked by the middleware are stored in a global variable called the *backlog*. By default, it holds up to 20 latest exchanges, but you can override by setting `HTTPOLICE_BACKLOG` to a different number.

The package also provides the `django_httpolice.report_view()` function. Add it to your URLconf like this:

```
import django_httpolice

urlpatterns = [
    # ...
    url(r'^httpolice/$', django_httpolice.report_view),
    # ...
]
```

When you start the server and open `/httpolice/` (or whatever URL you chose), you will see an HTML report on all the exchanges currently in the backlog. The **latest** exchanges are shown at the **bottom** of the report.

If `HTTPOLICE_ENABLE` is not `True`, the view responds with 404 (Not Found).

You can also access the backlog from your own code: it's in the `django_httpolice.backlog` variable, as a sequence of `httpolice.Exchange` objects. The latest exchange is `backlog[-1]`.

CHAPTER 4

Raising on notices

If you set the `HTTPOLICE_RAISE` setting to `'error'`, then the middleware will raise a `django_httpolice.ProtocolError` whenever a **response** is found to have any notices of severity "error" (that are not *silenced*). If you set it to `'comment'`, this will happen even for severity "comment".

The exchange is still added to the backlog.

This can be used to fail tests on problems:

```
$ python manage.py test
...E
======================================================================
ERROR: test_query_plain (example_app.test.ExampleTestCase)
----------------------------------------------------------------------
Traceback (most recent call last):
  [...]
  File "[...]/django_httpolice/middleware.py", line 92, in process_response
    raise ProtocolError(exchange)
django_httpolice.common.ProtocolError: HTTPolice found problems in this response:
------------ request: GET /api/v1/words/?query=er
C 1070 No User-Agent header
------------ response: 200 OK
E 1038 Bad JSON body


----------------------------------------------------------------------
Ran 4 tests in 0.380s

FAILED (errors=1)
```

CHAPTER 5

Silencing unwanted notices

To silence notices you don't care about, you can use the `HTTPOLICE_SILENCE` setting:

```
HTTPOLICE_SILENCE = [1070, 1110, 1194]
```

They will disappear from reports and will not cause `ProtocolError`.

By default, `HTTPOLICE_SILENCE` contains the following notices:

- 1110 because Django doesn't set `Date` on its responses, instead expecting the HTTP server to supply it
- 1070 because Django's test client doesn't send `User-Agent`

Of course, the `HTTPolice-Silence` header works, too:

```
def test_unauthorized(self):
    response = self.client.get('/api/v1/products/',
                               HTTP_HTTPOLICE_SILENCE='1194 resp')
    self.assertEqual(response.status_code, 401)
```