Watson - CORS

Release 1.0.0

Jan 15, 2018

Contents

1 Testing **1**

2 Contributing **3**

3 Table of Contents **5**

3.1 Usage 5

3.2 Configuring CORS 6

3.3 Reference Library 8

CHAPTER 1

Testing

Watson can be tested with py.test. Simply activate your virtualenv and run `python setup.py test`.

CHAPTER 2

Contributing

If you would like to contribute to Watson, please feel free to issue a pull request via Github with the associated tests for your code. Your name will be added to the AUTHORS file under contributors.

CHAPTER 3

Table of Contents

3.1 Usage

There are several ways you can implement CORS support into your Watson project.

1. Managed via @decorators in your controllers
2. Managed via route definitions
3. Managed via event listeners

3.1.1 Default Configuration

These configuration options are passed by default into your decorators or listeners.

```
defaults = {
    'allow_origin': None,
    'allow_credentials': False,
    'allow_methods': (
        'DELETE',
        'GET',
        'OPTIONS',
        'PATCH',
        'POST',
        'PUT',
    ),
    'allow_headers': (
        'accept',
        'accept-encoding',
        'authorization',
        'content-type',
        'dnt',
        'origin',
        'user-agent',
```

```
        'x-csrftoken',
        'x-requested-with',
    ),
    'expose_headers': None,
    'max_age': 86400,
}
```

For acceptable values for each of these, please read https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS

3.1.2 Option Types

allow_origin list|string: The allowed origin(s) for the request

allow_credentials boolean: Whether or not credentials are allowed

allow_methods list: The HTTP request methods that are allowed

allow_headers list: The HTTP headers that are allowed

expose_headers list: The HTTP headers which should be exposed

max_age integer: The length of time in seconds the OPTIONS response can be cached

3.1.3 Configuration Fallbacks

Whenever the listener or decorator is invoked, the following order of inheritance of options will be used when processing the request:

1. The default configuration (*watson.cors.config.defaults*)
2. The application configuration
3. **The route configuration (specify the arguments in the *options* property)**
4. The decorator arguments

3.1.4 Dynamic Allowed Origins

The preferred way to approach this issue would be to subclass *watson.cors.listeners.Request* and then modify the arguments that are sent through to *watson.cors.utils.process_cors*.

3.2 Configuring CORS

In addition to the individual ways to configure CORS, you can also take advantage of the way the configuration is inherited, and define common values in your application configuration.

```
# config.py

cors = {
    'allow_origin': 'http://127.0.0.1',
    'allow_credentials': True
}
```

3.2.1 Via Decorators

```
# controllers.py

from watson.cors.decorators import cors
from watson.framework import controllers
from watson.framework.views.decorators import view

class MyController(controllers.Rest):

    @view(format='json')
    @cors(allow_origin='http://127.0.0.1', allow_credentials=True)
    def GET(self):
        pass

    @cors(allow_origin='http://127.0.0.1', allow_credentials=True)
    def OPTIONS(self):
        return self.response
```

If you have already set the options in your application configuration, you can simply decorate the method with *@cors*.

3.2.2 Via Event Listener

```
# config.py

cors = {
    'allow_origin': 'http://127.0.0.1',
    'allow_credentials': True
}

events = {
    events.DISPATCH_EXECUTE: [
        ('watson.cors.listeners.Request',)
    ]
}
```

3.2.3 Via Routes

Routes modify the *allow_methods* option in a slightly different way to the other ways of configuring CORS. Any items set in the *accepts* property will override *allow_methods*.

```
# config.py

routes {
    'allow_origin': 'http://127.0.0.1',
    'allow_credentials': True
}

events = {
    events.DISPATCH_EXECUTE: [
        ('watson.cors.listeners.Request',)
    ]
}

routes = {
```

```
    'cors-request': {
        'path': '/cors',
        'options': {
            'controller': 'app.controllers.CorsRequest',
            'cors': {
                'expose_headers': ('my-custom-header',)
            }
        },
        'accepts': ('GET', 'OPTIONS')
    },
}
```

3.3 Reference Library