CMPUT 404 Lab 7

October 19, 2016

Overview

- Learn how to create a RESTful web application back-end using Flask.
- Set up Heroku with a basic Django application.
- Research issues concerning client-side scripting.

Steps

Flask

1. Navigate to a new folder and initialize a new Python virtual environment.

```
mkdir lab7
cd lab7
virtualenv venv
source venv/bin/activate
```

2. Install Flask.

```
pip install Flask
```

3. Create a new Python file named `hello.py` and edit its contents so it looks like this:

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello():
    return "Hello, World!"

if __name__ == "__main__":
    app.run(debug=True)
```

4. Run the application and navigate to the page in your browser.

```
python hello.py
```

5. Navigate back to your terminal and quite the running Flask application. Install the Python package `flask-restful`

```
pip install flask-restful
```

6. Open up `hello.py` and modify the file so it looks like the following:

```
from flask import Flask
from flask_restful import Resource, Api

app = Flask(__name__)
api = Api(app)

class HelloWorld(Resource):
    def get(self):
        return {'hello': 'world'}

api.add_resource(HelloWorld, "/")


if __name__ == "__main__":
    app.run(debug=True)
```

7. Try it in your browser or using cURL.

```
curl localhost:5000 # the port may be different on your machine
```

8. Let's implement a fully RESTful application for TODOs. Open up `hello.py` and add the following:

```
from flask import Flask
from flask_restful import reqparse, abort, Api, Resource

app = Flask(__name__)
api = Api(app)

parser = reqparse.RequestParser()
parser.add_argument('task')

# The latest NoSQL key-value store trending on Hacker News.
TODOs = {
    1: {'task': 'build an API'},
    2: {'task': '????'},
    3: {'task': 'profit'},
}

def abort_if_todo_not_found(todo_id):
    if todo_id not in TODOs:
        abort(404, message="TODO {} does not exist".format(todo_id))

def add_todo(todo_id):
    args = parser.parse_args()
    todo = {'task': args['task']}
    TODOs[todo_id] = todo
    return todo

class Todo(Resource):
    """
    Shows a single TODO item and lets you delete a TODO item.
    """

    def get(self, todo_id):
        abort_if_todo_not_found(todo_id)
        return TODOs[todo_id]

    def delete(self, todo_id):
        abort_if_todo_not_found(todo_id)
        del TODOs[todo_id]
        return '', 204

    def put(self, todo_id):
        return add_todo(todo_id), 201

class TodoList(Resource):
    """
    Shows a list of all TODOs and lets you POST to add new tasks.
    """

    def get(self):
        return TODOs

    def post(self):
        todo_id = max(TODOs.keys()) + 1
        return add_todo(todo_id), 201

api.add_resource(Todo, '/todos/<int:todo_id>')
api.add_resource(TodoList, '/todos')

if __name__ == '__main__':
    app.run(debug=True)
```

9. What does the browser show you when you navigate to these pages? Try out the following cURL commands:

```
curl localhost:5000/todos
curl localhost:5000/todos/3
curl -v -X DELETE localhost:5000/todos/2
curl -v -X POST localhost:5000/todos -d "task=something new"
curl -v -X PUT localhost:5000/todos/3 -d "task=something different"
```

Heroku

Please follow this guide to set up Django on Heroku:

https://devcenter.heroku.com/articles/getting-started-with-python#introduction

Protip:

HTTPie is like cURL, but designed specifically for interacting with RESTful JSON APIs. It colours output and pretty prints JSON automatically.

```
pip install httpie
http localhost:5000/todos
http :5000/todos
http HEAD :5000/todos
http POST :5000/todos task='try httpie!'
```

Questions

1. What does REST stand for? What does it mean?
2. What does CRUD stand for? For each letter in CRUD, give the associated HTTP method.
3. In general, what do HTTP 1xx status codes mean? HTTP 2xx? HTTP 3xx? HTTP 4xx? HTTP 5xx?
4. What is an XSS attack? Name one way a site can be vulnerable to an XSS attack.
5. What does CORS stand for? Under what situation in web application development will you need to care about implementing CORS? (Hint: What does the “CO” part of “CORS” mean?)