Flask RESTful Tutorial

Anyfactor

Chapter 3 Your First REST API

Starter App

This is standard boilerplate code for `Flask` which uses python decorators as routes to files.

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def home():
  return "hello world"

app.run(port=5000, debug=True)
```

Using JSON. One step closer to REST API

```
from flask import Flask, jsonify, request

# Request enables to capture POST requests.
# We don't need it for GET request as Flask does all that on its own

app = Flask(__name__)

# This is the database
stores = [
  {
    'name': 'store 1',
    items: [
      {
        'name': 'My Item',
        'price': 14.99
      }
    ]
  }
]

# Declare the method in the decorator
@app.route('/store', methods=['POST'])
def create_store():
  request_data = request.get_json()
```

```
  # This converts the JSON to python dictionary

  new_store = {
    'name': request_data['name'],
    'items': []
  }
  stores.append(new_store)
  # Appending to the original stores list
  return jsonify(new_store)
  # we are returning the store to tell the request was successful

@app.route('/store')
def get_stores():
# GET /store
# gets all the stores
# json requires json to be be sent
# But stores is a list; So we need to convert it to JSON/Dictionary
  return jsonify({'stores': stores})


@app.route('/store/<string:name>')
def get_store(name):
# GET request /store/store 1
# Gets specific store
# uses URL parameter and assigns the variable "name"
  for store in stores:
    if store['name'] == name:
      # name is from string:name
      return jsonify(store)
      # store is a dictionary already
  return jsonify({'message': 'store not found'})
  pass

@app.route('/store/<string:name>/item')
def get_item_store(name):
  for store in stores:
    if store['name'] == name:
      return jsonify({'items': stores['items']})
      # store items is a list
  return jsonify({'message': 'store not found'})

@app.route('/store/<string:name>/item', methods=['POST'])
def create_item_store(name):
  request_data = request.get_json()
  for store in stores:
    if store['name'] == name:
      new_item = {
        'name': request_data['name'],
        'price': request_data['price']
      }
      store['items'].append(new_item)
      return jsonify(new_item)
```

```
  return jsonify({'message': 'store not found'})


app.run(port=5000, debug=True)
```

Sending JSON with Postman

Select `create` then on the `headers`

key	Value
Content-Type	application/json

Select `body` and select `raw`

```
{
  "name": "another store"
}
```

Chapter 4 Flask RESTful

Setting up Virtual Environment

`pip install virtualenv`

`virtualenv venv --python=python3.8`

`venv` is the directory name

`python3.8` is declaration of the python version to be used

To activate the `venv` » `./venv/Scripts/activate.bat`

To exit/deactivate the `venv` » `deactivate`

`pip freeze` lets us see the modules in the venv

Preferably create a separate directory and work in it.