

REST API Basics

An introduction to RESTful web APIs

Inland Empire Python Users Group
June 21, 2022

John Sheehan

REST API Basics - Video

REpresentational State Transfer Application Programming Interface

A software *architecture* for creating web services (*Unlike SOAP which is a protocol*)

- Client-server architecture (*client makes requests for resources on server*)
- Stateless (*not dependent on user sessions*)
- Cacheable (*content is usually deterministic*)
- Layerable (*allows for load balancing and proxy servers*)
- Idempotent (*state does not change with multiple calls*)

- A request returns a resource representation based on the state provided by the request.
- The actual API is determined by the developer

Using RESTful APIs

An API request is made by the client via HTTP

A response is provided by the server typically in the form of JSON

```
import requests

def get_joke():
    endpoint = "https://karljoke.herokuapp.com/jokes"
    joke_type = "/programming"
    joke_qty = "/random"

    joke = requests.get(endpoint + joke_type + joke_qty).json()
    return joke[0]["setup"], joke[0]["punchline"]

if __name__ == "__main__":
    print(get_joke())
```

Using RESTful APIs

Some basic REST API guidelines:

- The base URL is considered to be the *endpoint* of the API
- RESTful APIs use nouns instead of verbs
- HTTP verbs are used for actions (POST, GET, PUT, DELETE)
- These map conveniently to CRUD operations (Create, Read, Update, Delete)
- Not all web APIs are RESTful
- When designing a RESTful API, consistency is important
- Most REST APIs require some type of authentication, often in the form of an API key

Example endpoint:

`https://api.mydomain.com/companystuff`

Example requests:

A GET request to `/user/` returns a list of registered users on a system

A POST request to `/user/123` creates a user with ID 123 using the body data

A PUT request to `/user/123` updates user 123 with the body data

A GET request to `/user/123` returns the details of user 123

A DELETE request to `/user/123` deletes user 123

Basic Routing

```
from flask import Flask, Response

app = Flask(__name__)

@app.route('/')
def index():
    return Response('Hello World!', 200)

if __name__ == "__main__":
    app.run(port=8080)
```

Basic Routing - JSON Response

```
from flask import Flask, jsonify, make_response

app = Flask(__name__)

@app.route('/')
def index():
    return jsonify({'data': 'Hello World!'})

@app.route('/add_customer')
def add_data():
    return make_response(jsonify({'result': 'customer added'}), 201)

if __name__ == "__main__":
    app.run(port=8080)
```


Basic Routing - HTTP Response Codes

HTTP status codes in a nutshell:

1xx Hold on.

2xx Here you go.

3xx Go away.

4xx You screwed up.

5xx I screwed up.

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>

REST API - URI Based

```
from flask import Flask, jsonify

app = Flask(__name__)

@app.route('/')
def index():
    return jsonify({'data': 'Hello World!'})

@app.route('/square/<int:num>', methods=['GET'])
def get_square(num):
    return jsonify({'result': num**2})

if __name__ == "__main__":
    app.run(port=8080)
```


REST API - Parameter Based

```
from flask import Flask, jsonify, request

app = Flask(__name__)

@app.route('/')
def index():
    return jsonify({'data': 'Hello World!'})

@app.route('/square', methods=['GET'])
def get_square2():
    num = int(request.args.get('num'))
    return jsonify({'args_result': num**2})

if __name__ == "__main__":
    app.run(port=8080)
```

Flask-RESTful

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

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

class Square2(Resource):
    def get(self):
        parser = reqparse.RequestParser()
        parser.add_argument('num', type=int, help='Number to square')
        args = parser.parse_args()
        num = int(args.get('num'))
        return {'args_result': num**2}

api.add_resource(Square2, '/square')

if __name__ == "__main__":
    app.run(port=8080)
```

Resources

- YouTube - REST API - Introducing REST :

<https://www.youtube.com/watch?v=HeXO98sogs8>

- Flask-RESTful Quickstart:

<https://flask-restful.readthedocs.io/en/latest/quickstart.html>

- Web API Directories:

<https://www.programmableweb.com/apis/directory>

<https://api.data.gov>

- Official Joke API:

https://github.com/15Dkatz/official_joke_api

- Source Code:

https://github.com/IEPUG/2020_02_REST-API

Discussion