## #14: API Programming with Flask

Computer Systems | March 7, 2023 · Wade Fagen-Ulmschneider

## **Sending HTTP Requests:**

In Python, the requests library provides us the ability to make HTTP requests to external APIs:

```
14/api.py
 1 import requests
 2
   r = requests.get("https://www.colr.org/json/color/random")
   print(f"Status Code: {r.status_code}")
   print(f"Character Encoding: {r.encoding}")
```

- requests.get(...) sends a GET request,
- requests.post(...) sends a POST request,
- requests.put(...) sends a PUT request,
- ...etc...

The requests library is just a wrapper around the request and response from any HTTP web service:

```
14/api.py
 7 print("== Headers ==")
 8 for header in r.headers:
      print(header + ": " + r.headers[header])
10
   print("== Payload (text) ==")
12 print(r.text)
13
14 print("== Payload (json) ==")
15 data = r.json()
16 print(data["colors"][0]["hex"])
```

## Note that:

- r.text returns the response as a string (at attribute).
- r. json() parses it for us into a dictionary for us to index into quickly (it's a function, requires the parameters)!

## **Receiving HTTP Requests:**

The flask library allows us to receive HTTP requests:

```
14/app.py
  1 from flask import Flask
  2 app = Flask(__name__)
    @app.route('/', methods=["GET"])
  5 def index():
      return "index function!"
    @app.route('/', methods=["POST"])
    def post():
     return "post function!"
 11
 12 @app.route('/hello', methods=["GET"])
 13 def hello():
      return "hello function!"
 14
 15
 16 @app.route('/hello/<id>')
 17 def with_id(id):
      return f"with_id function: {id}"
 18
 19
 20 @app.route('/hello')
 21 def mystery():
     return "mystery function!"
```

What happens with the following requests:

- 1. GET /
- 2. POST /
- 3. PUT /
- 4. GET /hello/
- 5. GET /hello
- 6. POST /hello
- 7. PUT /hello
- 8. GET /hello/42
- GET /hello/world