CS 340
Computer Systems

#15: API Programming with Flask

Oct. 12, 2023 · G Carl Evansr

Sending HTTP Requests:

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

14/api.py

```python
import requests

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

```python
print("== Headers ==")
for header in r.headers:
  print(header + ": " + r.headers[header])

print("== Payload (text) ==")
print(r.text)

print("== Payload (json) ==")
data = r.json()
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

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

@app.route('/', methods=["GET"])
def index():
  return "index function!"

@app.route('/', methods=["POST"])
def post():
  return "post function!"

@app.route('/hello', methods=["GET"])
def hello():
  return "hello function!"

@app.route('/hello/<id>')
def with_id(id):
  return f"with_id function: {id}"

@app.route('/hello')
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**
9. **GET /hello/world**