LECTURE 35

HTTP AND FLASK

MCS 275 Spring 2021

Emily Dumas

LECTURE 35: HTTP AND FLASK

Course bulletins:

- Please install **Flask**, e.g. with

  ```
  python3 -m pip install Flask
  ```

  in preparation for using it in upcoming assignments.

- If you already saw Flask, HTTP, CSS in MCS 260: Great, but don't get complacent! Project 4 will focus on database and web stuff.

MOCKUPS

First, let's check in on front page mockups for our two apps (chat and vote) with updated CSS.

Reminder: You can always get the code from the sample code repository.

FILE PROTOCOL

So far, I've been opening files in the web browser, using URLs with the `file` protocol.

There's no network communication here. The browser just opens the file using the OS interface.

To make an actual web site or application, we need an HTTP server.

PYTHON'S BUILT-IN HTTP SERVER

```
python3 -m http.server
```

Opens a web server that serves files in the current directory and its subdirectories.

Visit `http://localhost:8000/` in a browser (or substitute other port number shown in startup message) to see `index.html`.

Firewall rules typically prevent incoming connections from the internet (and maybe the local network too). That's good! Or

```
python3 -m http.server --bind 127.0.0.1
```

will make sure it **only** listens for connections from the same machine.

INDEX.HTML

Most HTTP servers that deliver resources from a filesystem will look for a file called `index.html` and send it in response to a request that ends in a `/`.

(i.e. if no filename is given, `index.html` is used.)

HTTP VERBS

- **GET** — Ask the server for a resource.
- **POST** — Submit data to a resource.

e.g. `GET /teaching/2020/fall/mcs260/` is sent to `dumas.io` when you load the home page of my Fall 2020 MCS 260 course.

More detailed look at an HTTP GET request: MCS 260 Fall 2020 Lecture 33.

HTTP RESPONSE CODES

The answer to any HTTP request includes a numeric code indicating success or error.

There are lots of codes; the first digit is often all you need to know:

- 2xx — success
- 3xx — redirection; more action required (e.g. moved)
- 4xx — client error; problem, your fault
- 5xx — server error; problem, not your fault

FLASK

Flask is a Python **web framework**. It makes it easy to write Python programs that respond to HTTP requests (e.g. web applications, APIs).

Competitors include:

- Bottle — minimalist like Flask
- Django — huge and full-featured

MINIMAL FLASK

```
from flask import Flask

app = Flask(__name__)

@app.route("/positivity/")
def name_of_function_does_not_matter():
    return """<!doctype html>
    <html>
        <body>
            You can do it!
        </body>
    </html>
    """

app.run()
```

REFERENCES

- jsfiddle - Write and test HTML+CSS quickly in browser
- HTML tutorial from w3schools
- CSS tutorial from w3schools
- The Flask tutorial

REVISION HISTORY

- 2021-04-09 Initial publication