Week 12 Lecture:
Python Web Programming (kind of)

Adam Hartz
hz@mit.edu

29 November 2021
What Is a Web Application?

Example: CAT-SOOP: collect and assess online exercises
Written in Python, 2011-now

Based on a program called “the tutor” by Tomás Lozano-Pérez (written in the Scheme dialect of LISP!)

Structurally, CAT-SOOP is a fairly standard web application. What does it do?

- **Receive request from user**
  (show me a page, submit this answer, etc)
- **Find relevant information**
  (page content, user info, history of submissions, etc)
- **Log new information if necessary**, and
- **Send response**
  (typically, HTML to be displayed in the browser)
The World Wide Web
3) IMAGE FILTERING VIA PER-PIXEL TRANSFORMATIONS

As our first task is manipulating images, we will look at an inversion filter, which reflects pixels about the middle gray value (i.e., black becomes white and vice versa). For example, here is a photograph of Adam’s cat. On the left side is the original image, and on the right is the inverted version.

Most of the implementation of the inversion filter has been completed for you. It is invoked by calling the method called `inversion`. However, some pieces have not been implemented correctly. Your task in this part of the lab is to fix the implementation of the inversion filter.

Before you do that, however, let’s add a simple test case so that we can test whether our code is working.

Let’s start with a 1x1 image that is defined with the following parameters:

- **height**: 1
- **width**: 1
- **pixels**: 0 (black, 1) (white)

**equation python literal**

If we were to run this image through a working inversion filter, what would the expected output be? In the box below, enter a Python list representing the expected `pixels` key in the resulting image.

```python
pixels = [255 for i in range(0, len(pixels))]
```

If we run our inversion filter, we might see:

```
> pixels = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

If we were to run this image through a working inversion filter, what would the expected output be? In the box below, enter a Python list representing the expected `pixels` key in the resulting image.

### Subsection: Adding a Test Case

Let’s also add this test case to the Lab’s regular tests so that it is run when we validate tests. If you open “test.py” in a text editor, you will see that:

```
<0> <0> import unittest
<0> <0> class TestInversion(unittest.TestCase):
<0> <0> def test_inversion(self):
<0> <0> <0> pass
```

Each class in this file serves as a collection of test cases, and each method within a class represents a particular test case.

```python
<0> <0> class TestInversion(unittest.TestCase):
<0> <0> <0> def test_inversion(self):
<0> <0> <0> <0> import json
<0> <0> <0> <0> <0> file = open("test_inversion.js", "w")
<0> <0> <0> <0> <0> file.write(json.dumps({
<0> <0> <0> <0> <0> <0> "test": {
<0> <0> <0> <0> <0> <0> <0> "name": "Another Test Case",
<0> <0> <0> <0> <0> <0> <0> "test": "test_inversion",
<0> <0> <0> <0> <0> <0> <0> "status": "success",
<0> <0> <0> <0> <0> <0> <0> "message": "Another Test Case passed successfully",
<0> <0> <0> <0> <0> <0> <0> "output": "success"
<0> <0> <0> <0> <0> <0> }
<0> <0> <0> <0> <0> })
```

```
<0> <0> file.close()
```

```python
3.11 Addline a Test Case

As staff, you are always allowed to submit. If you were a student, you would see the following:

You have submitted this assignment 0 times.
What Is a Server?
What Is a Server?
**Sockets**

*Sockets* allow communication across processes (on the same machine or different machines).

Typically, a *server* will wait for a *client* to make a connection on a designated *port* (a virtual endpoint for a connection).

Once the client connects, the socket allows for communication between the server and the client.

Client and server can each send/receive data via the socket.

Example: yelling echo server.
The Rest of Today

Designing a:

- Web Server
- Web Application Framework
- Web Application
HTTP Request and Response

GET /doc/test.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35

bookId=12345&author=Tan+Ah+Teck

HTTP/1.1 200 OK
Date: Sun, 08 Feb xxxx 01:11:12 GMT
Server: Apache/1.3.29 (Win32)
Last-Modified: Sat, 07 Feb xxxx
ETag: "0-23-4024c3a5"
Accept-Ranges: bytes
Content-Length: 35
Connection: close
Content-Type: text/html

<h1>My Home page</h1>

https://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html