

6.009: Fundamentals of Programming

Week 12 Lecture:

Python, Sockets, and the Web

Adam Hartz

hz@mit.edu

30 November 2020

What Is a Web Application?

Example: CAT-SOOP: collect and assess online exercises

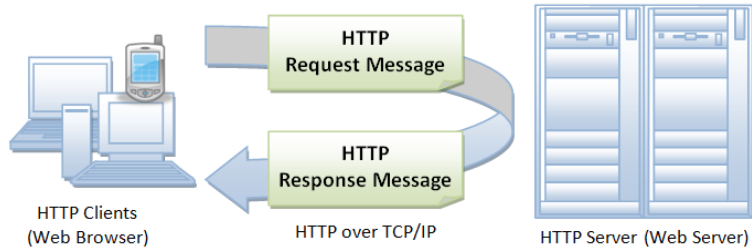
Written in Python, 2011-now

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



```

136 '255' white and vice versa). For example, here is a photograph of Adam H's cat. On
137 the left side is the original image, and on the right is an inverted version.
138
139 <center>
140 [stronger_the_cat] (CURRENT/stronger.jpg)
141 </center>
142
143
144 Most of the implementation of the inversion filter has been completed for you
145 (it is invoked by calling the method called 'inverted'), but some pieces have
146 not been implemented correctly. Your task in this part of the lab is to fix
147 the implementation of the inversion filter.
148
149 Before you do that, however, let's add a simple test case so that we can test
150 whether our code is working.
151
152 Let's start with a 1x4 image that is defined with the following parameters:
153
154 <python>
155 tutor.init_random()
156 img1_lo = cs.random.randint(0, 32)
157 img1_mid = cs.random.randint(64, 96)
158 img1_mid2 = cs.random.randint(128, 160)
159 img1_high = cs.random.randint(192, 224)
160 img1_pixels = [img1_lo, img1_mid, img1_mid2, img1_high]
161 </python>
162
163 • height: '1'
164 • width: '4'
165 • pixels: '9(img1_pixels)'
166
167
168 <question python:literal>
169 cs.prompts:
170 If we were to run this image through a working inversion filter, what would the
171 expected output be? In the box below, enter a Python list representing the
172 expected 'pixels' key in the resulting image.<br/>
173 ...
174
175 cs.soln = ['255-i for i in img1_pixels]
176 def cs.check_function(sub, sol):
177     if not(isinstance(sub, list)):
178         return (0.0, 'Please enter a Python list.')

```

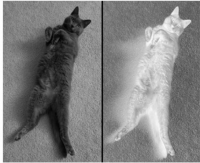
10 Haste The Day - Su
6.009 Fall 2017

https://eecs6009.mit.edu/fall17/abs/lab1
90%

6.009
Home / Lab 1 Image Processing
Homepage
Information
Quizzes
Labs
Help
Log Out

3) IMAGE FILTERING VIA PER-PIXEL TRANSFORMATIONS

As our first task in manipulating images, we will look at an inversion filter, which reflects pixels about the middle gray value (0 black becomes 255 white and vice versa). For example, here is a photograph of Adam H's cat. On the left side is the original image, and on the right is an inverted version.



Most of the implementation of the inversion filter has been completed for you (it is invoked by calling the method called `inverted`), but 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 1x4 image that is defined with the following parameters:

- height: 1
- width: 4
- pixels: [12, 76, 154, 222]

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:

Save
Submit
View Answer

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.

3.1) Adding a Test Case

What Is a Server?



What Is a Server?



Sockets

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

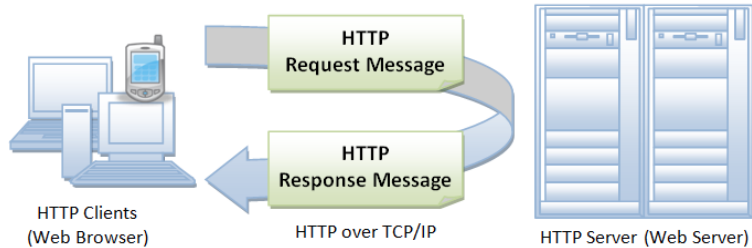
Typically, a *server* will wait for a *client* to make a connection on a designated *port* (a virtual endpoint for a connection).

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 World Wide Web



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
```

Annotations for Request:

- Request Line: `GET /doc/test.html HTTP/1.1`
- Request Headers: `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`
- A blank line separates header & body
- Request Message Body: `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>
```

Annotations for Response:

- Status Line: `HTTP/1.1 200 OK`
- Response Headers: `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`
- A blank line separates header & body
- Response Message Body: `<h1>My Home page</h1>`

The Rest of Today

Designing a:

- Web Server
- Web Application Framework
- Web Application