CMPSC 381
Data Communications and Networks
Spring 2016
Bob Roos

`http://cs.allegheny.edu/sites/rroos/cs381s2016`

Lab 2
4 February 2016
Due via Bitbucket on Thursday, 11 February, 8 a.m.
NOTE THE 8 a.m. DEADLINE!

Summary: Build your own Web server in Python.

Details:

1. **[Get files.]** On the course website, on the "Lab Handouts" page, find the files "`lab2skeleton.py`", "`favicon.ico`", "`page1.html`", and "`test.py`" and save them in the `lab2` directory of your bitbucket repository.

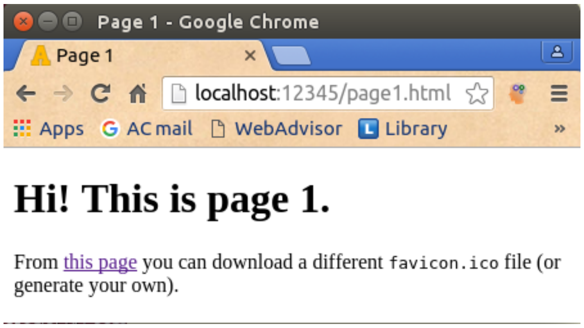
2. **[Make sure things work.]** Before you start modifying the Python program, make sure it works correctly. In your `lab2` directory run the command:

```
python lab2skeleton.py
```

Open Chrome and, in the browser window, type:

```
localhost:12345/page1.html
```

If things are working correctly, you should see something like this in your browser window:



Hi! This is page 1.

From this page you can download a different `favicon.ico` file (or generate your own).

3. **[Create a Better Web Server.]** Copy the `lab2skeleton.py` file into a new file named "*yourlastname*-`lab2.py`". Edit this file as follows: Check each of the following conditions to make sure they are met; if any one of them fails, your server should *not* return the requested file but should, instead, return an "Error 400 Bad Request:" message naming the specific error, for instance,

```
Error 400 Bad Request: missing GET
```
or
```
Error 400 Bad Request: missing Host header
```

I have done first one for you in the `lab2skeleton` file.

- [DONE!]The first line of the message must have exactly three fields separated by spaces (typically this will be something like "`GET /page1.html HTTP/1.1`").
- The first field must be the word "`GET`" in all uppercase letters
- The second line of the message must have exactly two fields separated by spaces; the first of these must be the string "`Host:`"

Other header fields will be ignored for now (but see the optional portion of the assignment)

4. [**Test your solution.**] In one terminal window, run your Web server; in another terminal window, run the file "`test.py`." This will make repeated attempts to connect to your server; each attempt has one of the errors listed above.

5. [**Add One Other Modification.**] Either add a new test (for instance, test to see that the correct hostname is supplied) or else expand the functionality (for instance, if the filename ends in a slash "`/`" then assume it is the name of a directory containing an "`index.html`" file and return that file). Describe in your program header comments what your modification does.

6. [**OPTIONAL—BUT FUN!**] There are various add-ons to Firefox and Chrome that permit you to add new headers to requests. For instance, in Chrome you can install the "Modify Headers" app from Rivet Systems, Inc. In Firefox you can install "Modify Headers" from Gareth Hunt. (I'd be happy to help you figure out one of these tools.) Now invent a new header—e.g., a "`Square-root:`" header that has a numeric value. Modify your web browser so that, when it sees a "`Square-root:`" header it returns a new header named "`Answer:`" that contains the square root of the given value.

   You can inspect the response headers in the "developer tools" of your chosen browser (ask—it's not obvious where to find them).

7. [**Submit your work.**] Make sure you have a webserver named "*yourlastname*`-lab2.py`" in your `lab2` folder. Upload this folder to your Bitbucket repository by the lab deadline.

   Make sure your name, and the honor code pledge appear in your header comments. Make sure you have commented all of your added code and deleted obsolete comments.

NOTES: If you want to stop your server manually you will need to kill it using the CTRL-C key.

It sometimes takes a few seconds for the port to be freed up when the server dies, so if you get a message saying "Address already in use," just wait a few seconds and try again.