Author: Jackson Lee

Date: 01/26/2023

Project: 1 – Sockets and HTTP

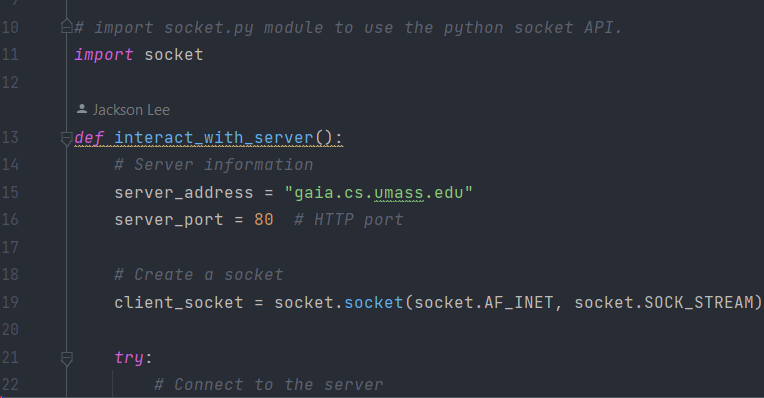
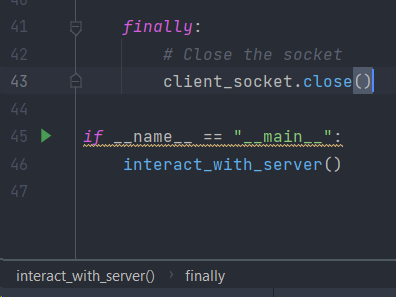
Course: CS 372 Winter 2024

Description: Create a simple python program that uses a socket to interact with a server.

INTRODUCTION

This project demonstrates how to make a bare-bones socket program to do the same thing as what my internet browser does. I will be using Python 3 as a programming language to send request by using a socket to interact with a file server, and to retrieve an arbitrarily large file.

PART 1: Using a socket to GET a file

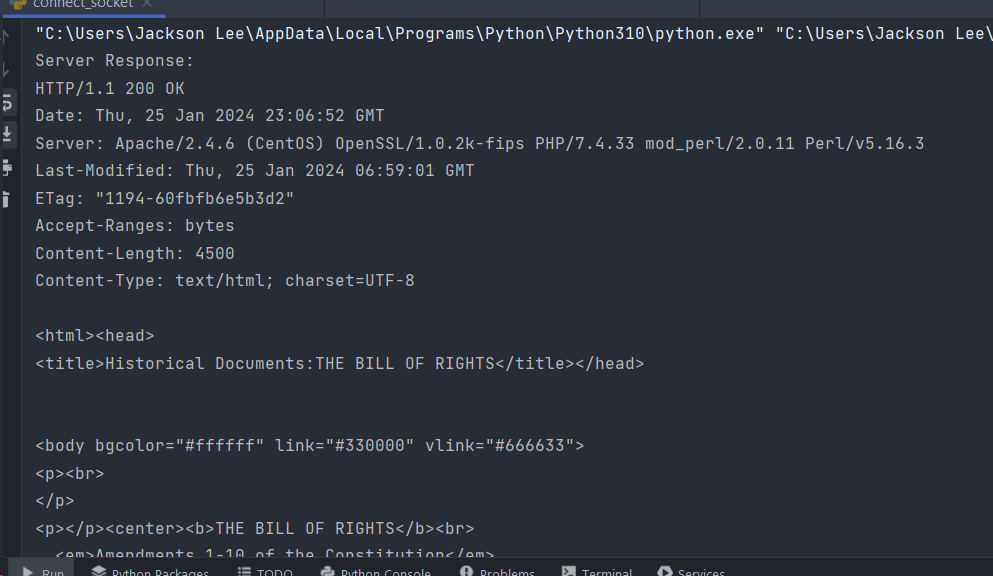
Part 1 was a practice exercise to run my program and receive a response from the server. I used Python 3 to write a small procedure to test the socket connectivity and collected a very small file. In my coding, I first imported the socket.py Python library module to use the socket API, then I drafted a procedure that interact with the server using the try method and finally method to help test the block of code for error and to ensure the open socket was properly closed upon completing the request. I used the coding format that were best practices taught from CS 161 and CS 162 courses from previous terms.   
  
Here are some screenshots of my running codes:   
  
 

PART 2: GET the data for a large file

INSTRUCTIONS FOR HOW TO RUN

1. Ensure Internet connectivity is currently active
2. Download and execute the latest version of PyCharm Community Edition
3. Open the file connect\_socket.py
4. (Shift + F10) or press the Run button to execute the code.
5. Review the expected results in the bottom panel in the Python Console.

SCREENSHOT 1 and 2: first few lines and last few lines of results as follow



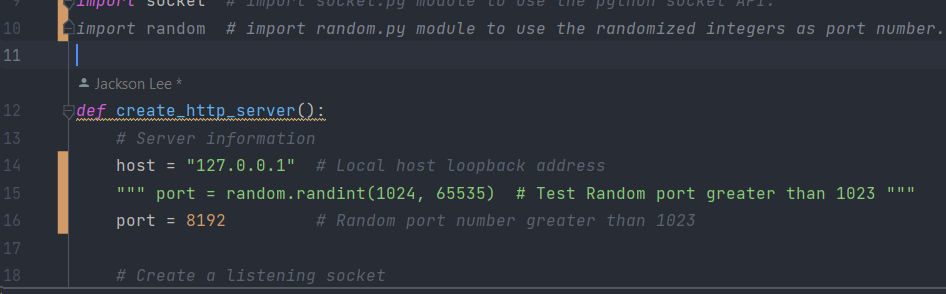
A screenshot of a computer

Description automatically generated

PART 3: The World’s simplest HTTP server

INTRODUCTION

This program utilizes the loopback address of any IP capable system, along with a randomized port number, to send request to itself and receive data. I tested using the python library random module to randomize the port number as a trial run, then selected a dedicated port number as 8472 (referencing Star Trek Voyager series for an advanced and powerful species that are immune to the Borg’s assimilation process!) for the final code. Here are some screenshots of my coding:

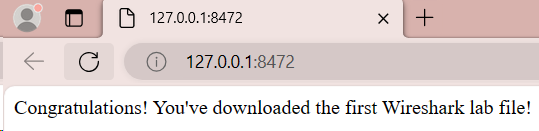
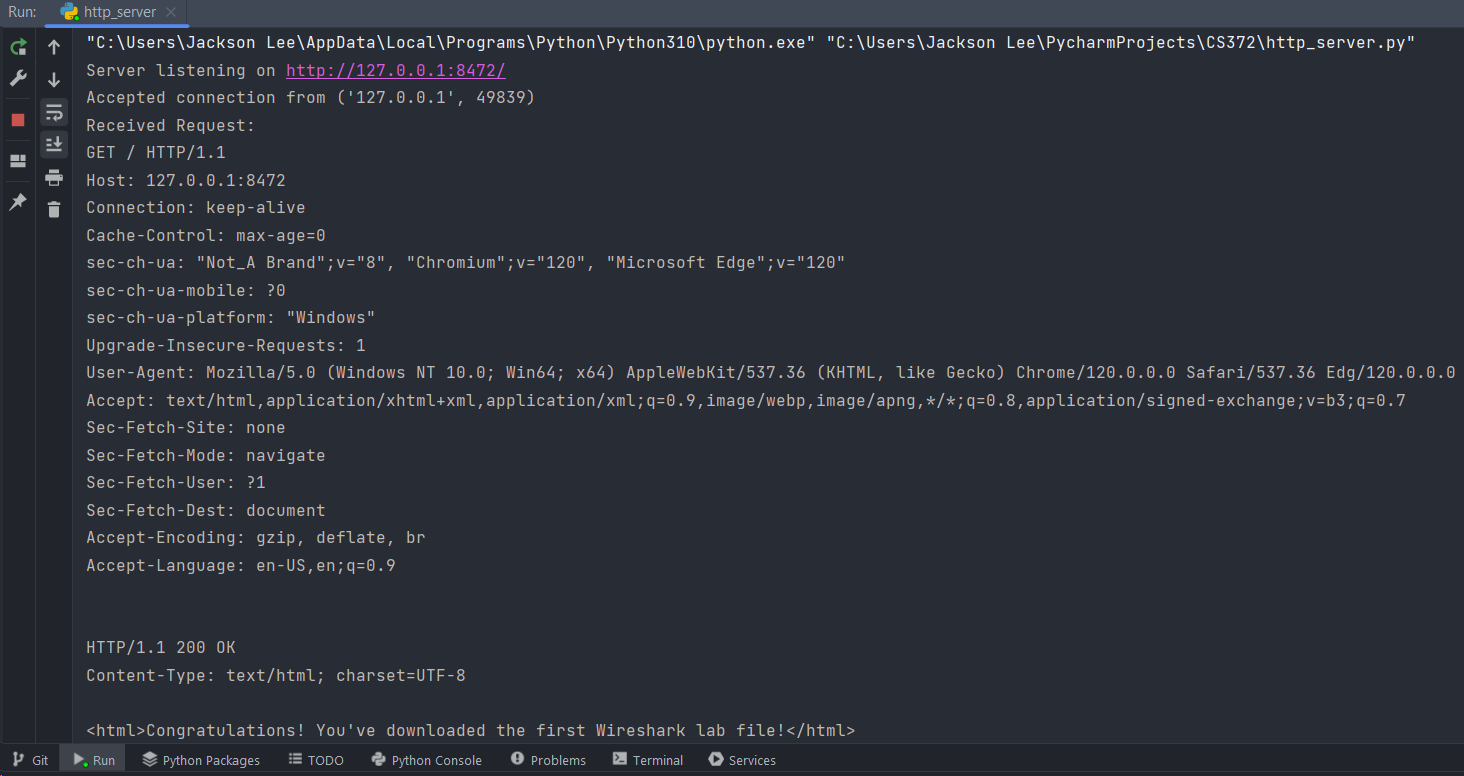


And again, I used the finally method to securely close the listening port toward the end of the coding.



INSTRUCTIONS FOR HOW TO RUN

1. Ensure Internet connectivity is currently active
2. Launch PyCharm Community Edition
3. Open the file http\_server.py
4. (Shift + F10) or click Run button to execute the code.
5. The Server should now be listening on a random port (numbers are being displayed).
6. Open a web browser on the same system where the code is being ran.
7. Navigate to 127.0.0.1:8472 (selected random port number).
8. Results from screenshot below should be expected:

COMMENTS

The reason I had to empathize that Internet connectivity on step 1 on each of the instructions for how to run my code is due to the fact that we are working with actual real servers on the Internet and only if there is live Internet access will the code be ran correctly with the expected results.

SOURCES CITED

1. Kurose, J. F., & Ross, K. W. (n.d.). Computer Networking: A Top-Down Approach (8th ed.). Pearson. (Chapters 2.2–2.4, 2.6, 2.7).
2. Python Software Foundation. (n.d.). Built-in Exceptions. Python 3.10.0 documentation. Retrieved January 25, 2024, from <https://docs.python.org/3/library/exceptions.html>
3. Python Software Foundation. (n.d.). Socket — Low-level networking interface. Python 3.10.0 documentation. Retrieved January 25, 2024, from <https://docs.python.org/3/library/socket.html>