UNIT- IV

Python to access Web Data : cont'd.

- Regular Expressions
- Extracting data
- Sockets
- Using the Developer Console to Explore HTTP
- Retrieving Web Page, and Passing Web Pages

Prof. R. MADANA MOHANA
Professor, Artificial Intelligence & Data Science
http://rmadanamohana.com/

Reference

https://www.coursera.org/learn/python-network-data

https://github.com/I-am-Harsh/Using-Python-to-Access-Web-Data

https://www.py4e.com/code3/

https://eng.libretexts.org/Bookshelves/Computer_Science/Programming_Languages/Book%3A_Python_for_Everybody_(Severance)

http://docs.python.org/library/socket.html

https://youtu.be/T0rYSFPAR0A

Networked Programs:

- Sockets

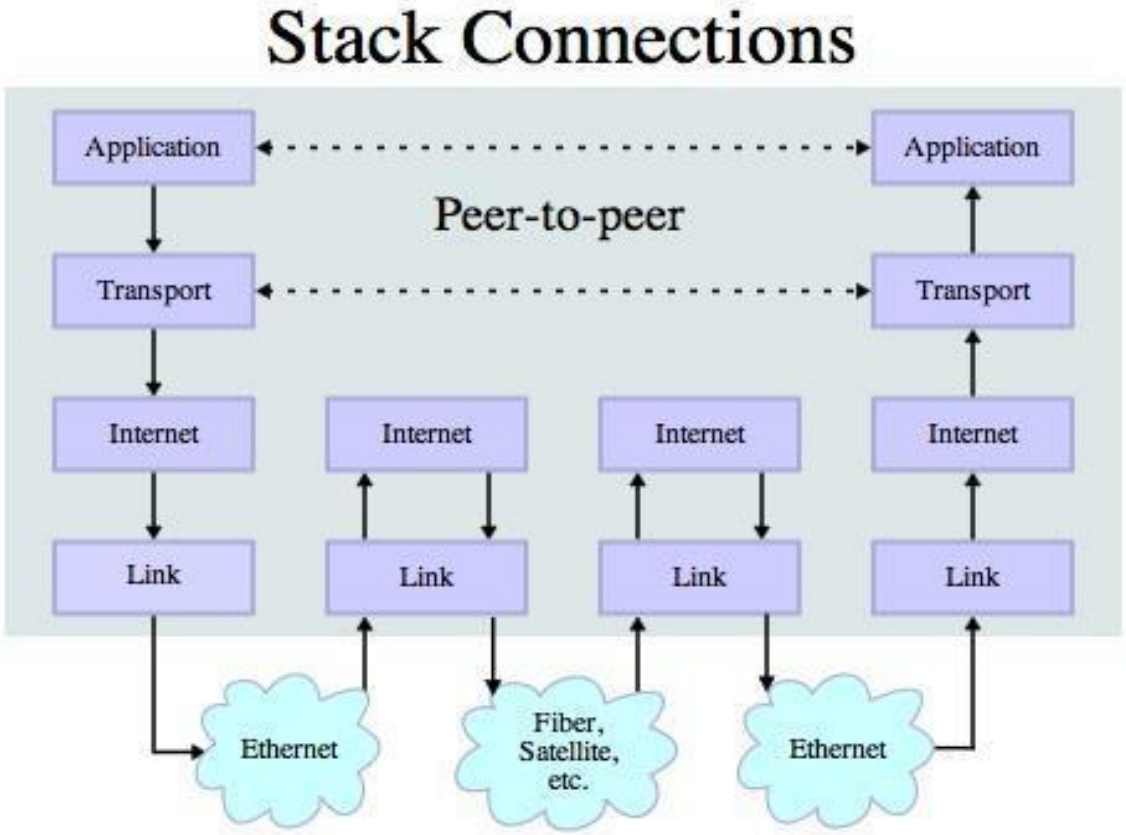
Transport Control Protocol (TCP)

- Built on top of IP (Internet Protocol)
- Assumes IP might lose some data - stores and retransmits data if it seems to be lost
- Handles “flow control” using a transmit window
- Provides a nice reliable channel

IP Address

- An IP (Internet Protocol) address is a numerical label assigned to each device (such as a computer or a smartphone) connected to a network
- It serves *two main purposes*:
  - ✓ Identifying the host or network interface and
  - ✓ Providing the location of the device in the network
- IPv4 (Internet Protocol version 4) addresses are the most common and consist of four sets of numbers separated by periods (e.g., 192.168.0.1)
- IPv6 (Internet Protocol version 6) addresses are the *newer version* and are represented as eight groups of hexadecimal digits separated by colons (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334)

Transport Control Protocol (TCP)



Socket

- A socket is an endpoint for communication between two processes over a computer network.
- Socket serves as Application Programming Interface (API) that allows processes to establish connections, send, and receive data.
- The concept of sockets is based on the client-server model, where one process acts as a server and listens for incoming connections, while other processes act as clients and establish connections to the server.
- Once a connection is established, processes can exchange data through the socket.

TCP Connections / Sockets

- "In computer networking, an Internet socket or network socket is an endpoint of a bidirectional inter-process communication flow across an Internet Protocol-based computer network, such as the Internet."
- A single network will have two sockets.
- Sockets are a combination of an IP address and a Port.

Sockets Terminology

Domain

- The family of protocols that is used as the transport mechanism.
- These values are constants such as AF_INET (Address Family Internet), PF_INET (Protocol Family Internet), PF_UNIX, PF_X25 (X.25 is an ITU-T (developed by Telecommunication Standardization Sector of International Telecommunication Union) standard protocol suite for packet-switched data communication in wide area networks), and so on.

Type

- The type of communications between the two endpoints, typically SOCK_STREAM (Stream (connection) Socket) for connection-oriented protocols and SOCK_DGRAM (Datagram (conn.less) Socket) for connectionless protocols.

Sockets Terminology

Protocol

- Typically zero, this may be used to identify a variant of a protocol within a domain and type.

hostname

- The identifier of a network interface:
  - ✓ A string, which can be a host name, a dotted-quad address, or an IPV6 address in colon (and possibly dot) notation
  - ✓ A string "<broadcast>", which specifies an INADDR_BROADCAST address (IPv4 broadcast address).
  - ✓ A zero-length string, which specifies INADDR_ANY (IPv4 local host address), or
  - ✓ An Integer, interpreted as a binary address in host byte order

Sockets Terminology

port

- Each server listens for clients calling on one or more ports.
- A port may be a Fixnum port number, a string containing a port number, or the name of a service.

Ports

- Ports are virtual endpoints for communication in a network
- A port is associated with an IP address and allows multiple applications to run simultaneously on a device
- Ports are identified by numbers ranging from 0 to 65535
- Well-known ports (0-1023) are assigned to specific services (e.g., HTTP uses port 80, HTTPS uses port 443)
- Registered ports (1024-49151) are assigned by the Internet Assigned Numbers Authority (IANA) for specific purposes
- Dynamic or private ports (49152-65535) can be used by applications dynamically or for private purposes

TCP Port Numbers

- A port is an application-specific or process-specific software communications endpoint
- It allows multiple networked applications to coexist on the same server
- There is a list of well-known TCP port numbers

Common TCP Port Numbers

Protocol	Port Number	Python Library	Function
HTTP (Hyper Text Transfer Protocol)	80	`httplib, urllib, xmlrpclib`	Web pages
FTP (File Transfer Protocol)	20	`ftplib, urllib`	File transfers
NNTP (Network News Transfer Protocol)	119	`nntplib`	Unsent news
SMTP (Simple Mail Transfer Protocol)	25	`smtplib`	Sending mail
Telenet	23	`Telnetlib`	Command lines
POP3 (Post Office Protocol)	110	`Poplib`	Fetching email
Gopher (It is an application-layer protocol that provides the ability to extract and view Web documents stored on remote Web servers)	70	`Gopherlib`	Document transfer

Python *socket* Module

Python has built-in support for TCP Sockets

First import the Python socket module for this

```
>>> import socket
```

To create a socket, we must use the `socket.socket()` function available in socket module, which has the following general syntax:

```
s = socket.socket (socket_family, socket_type, protocol=0)
```

Here is the description of the parameters:

`socket_family` – This is either AF_UNIX or AF_INET

`socket_type` – This is either SOCK_STREAM or SOCK_DGRAM

`protocol` – This is usually left out, defaulting to `0`

Important Methods of the Socket module

Methods	Description
`socket.socket()`	Used to create sockets (required on both server as well as client ends to create sockets)
`socket.accept()`	Used to accept a connection, It returns a pair if values (conn, address)
`socket.bind()`	Used to bind to the address that us specified a parameter
`socket.close()`	Used to bind the address that is specified as a parameter
`socket.connect()`	Used to mark the socket as closed
`socket.listen()`	Used to connect to a remote address specified as the parameter

Server and Clients

Server

- Either a program, a computer, or a device
- Dedicated to managing network resources
- Can be on the same device or computer or local or even remote
- There are various *types of servers* such as database server, network server, front-end server etc.
- Servers commonly make use of *methods* like socket methods, bind and listen methods etc. to establish a connection and bind to the client

Server and Clients

Client

- Computer or software that receives information or services
- Clients requests for services from servers
- Example: Web browsers such as Google Chrome, Firefox etc.
- These web browsers request web servers to send required web pages and services as directed by the user
- Other examples of clients include online games, online chats etc.

Python Program to demo Client-Server Communication

Server-side program:

File Name: sockets_server.py

```
import socket
myserver=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
# AF_INET referes the address from the Internet and
SOCK_STREAM is used to create TCP protocol
myserver.bind((socket.gethostname(), 1024))
#Here 1024 is the Port Number. Provide any Port number is
>1023 which is non previliged.
myserver.listen(5)
#This asks for permission on Windows
```

Python Program to demo Client-Server Communication

Server-side program: cont'd.

File Name: sockets_server.py

```
while True:
    myclient,addr=myserver.accept()
    print(f"Connection to {addr} established")
    # f string is literal string
    myclient.send(bytes("Welcome to Socket Programming in Python!", "utf-8"))
    # To pass the message from server to client. The kind of byte is utf-8 (Unicode Transformation Format – 8-bit)
```

Python Program to establish a connection between Server & Client

Client-side program:

File Name: sockets_client.py

```
import socket
myserver=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
myserver.connect((socket.gethostname(), 1024))
# 1024 is same port number given in server
msg=myserver.recv(1024)
# 1024 bytes is is just sample byte number. Received
message from Server is stored in "msg"
print(msg.decode("utf-8"))
# "utf-8" is bytes code given in server. Message received
from the Server can be decoded.
```

Python Program to establish a connection between Server & Client

Running Procedure:

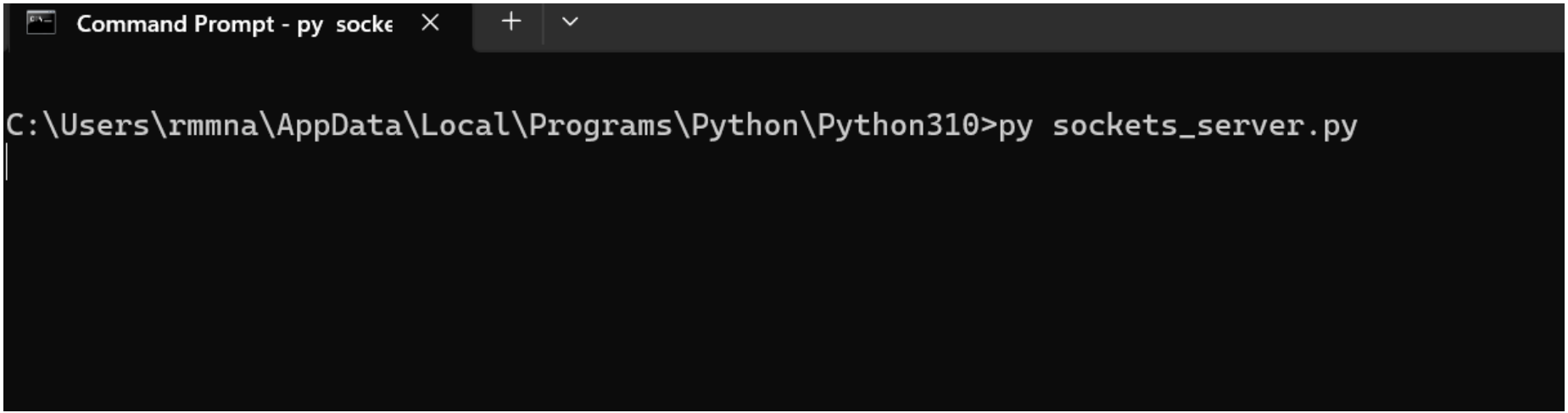
Step-1: Goto Command prompt and then change directory to where `sockets_server.py` and `sockets_client.py` programs are located

```
Microsoft Windows [Version 10.0.22621.1848]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rmmna>cd AppData

C:\Users\rmmna\AppData>cd Local

C:\Users\rmmna\AppData\Local>cd Programs

C:\Users\rmmna\AppData\Local\Programs>cd Python

C:\Users\rmmna\AppData\Local\Programs\Python>cd Python310

C:\Users\rmmna\AppData\Local\Programs\Python\Python310>dir *.py
 Volume in drive C is Windows
 Volume Serial Number is 4C6B-281F

 Directory of C:\Users\rmmna\AppData\Local\Programs\Python\Python310

25-04-2023  10:25               487 abstractclass.py
09-05-2023  21:07                97 arithmetic operators.py
20-04-2023  21:48                 2 classtype.py
20-04-2023  20:36               465 derivedclass.py
17-03-2023  22:36                60 ex.py
30-04-2023  21:56               771 exceptions with inheritance.py
06-07-2023  19:43               441 GREEDYEX.py
20-04-2023  19:17               467 overriding.py
09-05-2023  21:51                88 relational operators.py
25-04-2023  10:36               308 singletonclass.py
11-07-2023  15:02               175 sockets_client.py
11-07-2023  15:10               360 sockets_server.py
20-04-2023  20:07               599 typeconversions.py
```

Python Program to establish a connection between Server & Client

Running Procedure:

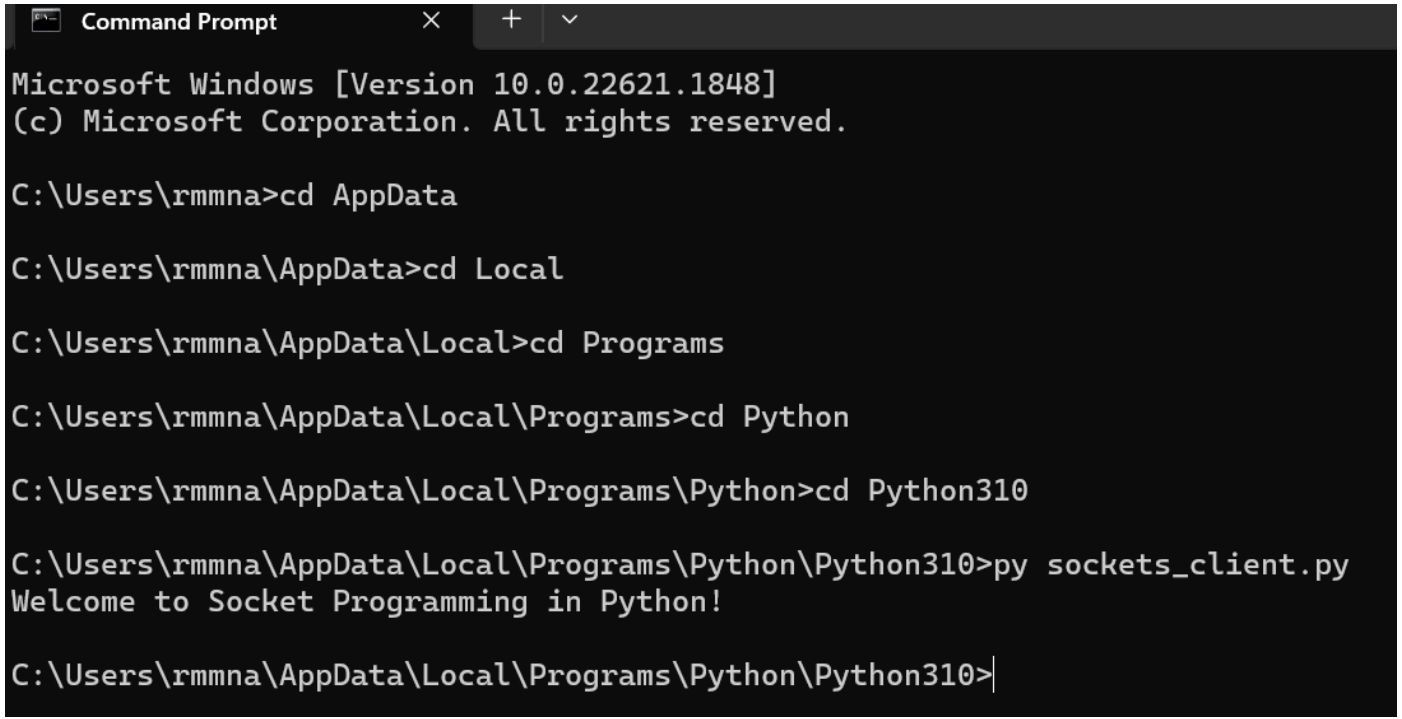
Step-2: Run the Server Program "`sockets_server.py`" using the following python command in command prompt and minimize this window.



Python Program to establish a connection between Server & Client

Running Procedure:

Step-3: Run the Client Program "`sockets_clients.py`" by opening the new command prompt window using the following python command.



Python Program to establish a connection between Server & Client

Running Procedure:

Step-4: Server shows connection establishment