**CN Lab**

## **What is Socket Programming?**

Sockets (aka socket programming) is a program that enables two sockets to send and receive data, bi-directionally, at any given moment.It works by connecting two sockets (or nodes) together and allowing them to communicate in real time, and is a great option for building a myriad of apps.

## **Why Use Sockets to Send Data?**

Internet-connected applications that need to operate in real time greatly benefit from the implementation of sockets in their networking code. Some examples of apps that use socket programming are:

* Web pages that show [live notifications](https://www.pubnub.com/use-case/alerts-and-push-notifications/) (Facebook, Twitch, eBay)
* [Multiplayer online games](https://www.pubnub.com/industry/gaming/) (League of Legends, WoW, Counter Strike)
* [Chat apps](https://www.pubnub.com/use-case/in-app-chat/) (WhatsApp, WeChat, Slack)
* [Real-time data dashboards](https://www.pubnub.com/use-case/data-broadcasting-and-dashboards/) (Robinhood, Coinbase)
* IoT devices (Nest, August Locks)

## **Python Socket Programming**

Python provides a [socket class](https://docs.python.org/3/library/socket.html) so developers can easily implement socket objects in their source code

### **Import Socket Library**

* 1. To use a socket object in your program, start off by importing the socket library. No need to install it with a package manager, it comes out of the box with Python.
  2. **Import socket**

1. **Build Socket Objects** 
   1. Now we can create socket objects in our code.
   2. **sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)**
   3. This code creates a socket object that we are storing in the “sock” variable. The constructor is provided a family and type parameter respectively. The family parameter is set to the default value, which is the Address Format Internet.  
      The type parameter is set to Socket Stream, also the default which enables “sequenced, reliable, two-way, connection-based byte streams” over TCP

### **Open and Close Connection**

* 1. Once we have an initialized socket object, we can use some methods to open a connection, send data, receive data, and finally close the connection.
  2. sock.connect(('0.0.0.0', 8080)) ## Connect to an IP with Port, could be a URL
  3. sock.send("Twenty-five bytes to send") ## Send some data, this method can be called multiple times
  4. sock.recv(4096) ## Receive up to 4096 bytes from a peer
  5. sock.close()## Close the socket connection, no more data transmission

**Complete Code**

## **Python Socket Client Server**

1. **import socket**
2. **serv = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)**
3. **serv.bind(('0.0.0.0', 8080))**
4. **serv.listen(5)**
5. **while True:**
6. **conn, addr = serv.accept()**
7. **from\_client = ''**
8. **while True:**
9. **data = conn.recv(4096)**
10. **if not data: break**
11. **from\_client += data**
12. **print from\_client**
13. **conn.send("I am SERVER<br>")**
14. **conn.close()**
15. **print ('client disconnected')**

This code makes a socket object, and binds it to localhost’s port 8080 as a socket server. When clients connect to this address with a socket connection, the server listens for data, and stores it in the “data” variable.Then, the program logs the client data using “print,” and then sends a string to the client: I am SERVER.

## **Python Socket Client**

1. **import** socket
2. client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)
3. client.connect(('0.0.0.0', 8080))
4. client.send("I am CLIENT<br>")
5. from\_server = client.recv(4096)
6. client.close()
7. **print** (from\_server)

This client opens up a socket connection with the server, but only if the server program is currently running. To test this out yourself, you will need to use 2 terminal windows at the same time.Next, the client sends some data to the server: I am CLIENT.Then the client receives some data it anticipates from the server.Done! You can now get started streaming data between clients and servers using some basic Python network programming.

|  |  |  |  |
| --- | --- | --- | --- |
| **Protocol** | **Common function** | **Port No** | **Python module** |
| HTTP | Web pages | 80 | httplib, urllib, xmlrpclib |
| NNTP | Usenet news | 119 | nntplib |
| FTP | File transfers | 20 | ftplib, urllib |
| SMTP | Sending email | 25 | smtplib |
| POP3 | Fetching email | 110 | poplib |
| IMAP4 | Fetching email | 143 | imaplib |
| Telnet | Command lines | 23 | telnetlib |
| Gopher | Document transfers | 70 | gopherlib, urllib |

**Task**

Write python program that creates three client and print message I am client 1 , I am client 2 and I am client 3 using client server architecture