Python 101

A Practical Introduction To Networks and HTTP, and Building a HTTP Server From Scratch

Contents

Multithreading

Threads are nothing but a light weight process, their creation and destruction is easy in memory. But as they are light weight they can not be assigned heavy tasks. There are two types of threads, Kernel Threads and User Threads. Kernel Threads are used by the Kernel of our OS and User threads are not a part of OS. In Python there are two modules that support multithreading namely **_thread** and **threading**. _thread is a deprecated module which is only kept for backward compatibility. It provides us functionality to convert a function into a thread whereas threading has a class named Thread which should be extended by another class to execute the class's run method in thread.

See example ex_threading

Introduction to Networks

We already know that if we need data transfer (communication) between any two endpoints we need some data-stream.

It seems trivial to understand how it might be happening when both endpoints are within our systems. But it becomes increasingly tough when the source and receiver are different systems.

Its obvious we need some medium to create the data stream. That medium through which we transfer data between different systems is called Network.

Internet is essentially an example of network on which all of our devices are connected.

Introduction to Networks (IP Address)

The component in our systems that is used to connect to various networks is called NIC (Network Interface Controller).

Now lets consider this case we have a network on which there are 10 devices connected we also connect to this network and now we want to send message to a specific device. How can we possibly do that? We need some unique identification for all of the devices so that we can send messages to a specific device.

This unique identification for all devices is called IP Address. If a device is connected to multiple networks it will have multiple IPs each corresponding to it's identity on that network.

Introduction to Networks (IP Address) contd..

There is an IP address called Loopback Address or Localhost.

localhost is the IP address **127.0.0.1** , This IP Address points to our own computer. As it points to our own computer it can be accessed irrespective of the fact that we are connected on a network or not.

Now why is this needed?

Initially when network engineers started programming for network based applications they felt why should they expose their application to an external network in order to **test** their code. They should be able to do it within their system. In order to accomplish this **localhost** was created.

Introduction to Networks (Ports) contd..

There are many services running on our systems at an instant on a network.

When a message arrives at our system How will our system define for which service this message is intended?

In order to do this we assign a unique number ranging from 0 to 65535 to each service.

Some of these numbers are reserved by our system for internal services.

Introduction to Networks (Server and Client) contd..

Server is simply the endpoint in communication that **accepts** the connection to establish a network and Client is the endpoint in communication that **asks** server to accept his connection.

Servers are also a socket on a device on internet with a certain IP Address just like our systems nothing more or less.

For example, Google accept our connection requests sent from our browsers so in that term browsers are clients and google and many other websites are all servers as they accept our connection.

Introduction to Networks (Sockets) contd..

Sockets are one endpoint in a data flow over a network.

We can implement them using **socket** module in python.

See example ex_sockets

Introduction to Networks (Protocol) contd..

We know that a protocol is a set of rules for communication.

HTTP is such a protocol for transferring data over World Wide Web.

All network essentially work on a set of protocols and all of the networks are divided in several layers based on functionality and other factors. This set of layers is known as IP Stack (Formerly known as OSI Stack)-

1. Data Link Protocols - eg. Ethernet, WiFi
2. Network Protocols - eg. IP, IPsec.
3. Transport Protocols - eg. TCP, UDP etc.
4. Application Protocols - eg. HTTP, HTTPs, FTP etc.

Introduction to Networks (Protocol) contd..

HTTP protocol works based on TCP Protocol.

The URL we enter in our browsers means-

protocol://host-name:port/path

HTTP Protocol's request messages follow this simple syntax-

Method path http_version

Methods are HTTP specific methods eg. GET, POST etc.

Introduction to Networks (Protocol) contd..

GET is most of the times used to get some result on client side. Where as POST is most of the times used to send something to the server side.

Path is nothing but a mapping on server side to various functionalities. Eg. if someone asks for /user path then i will return the profile, if someone asks for /dashboard i will return the dashboard etc.

HTTP version stands for the HTTP Protocol that we are using, currently HTTP/2.0 is latest but we will be implementing HTTP/1.1 which is still widely being used.

Introduction to Networks (Protocol) contd..

Similarly HTTP Response format is as follows-

http_version response_code response_code_explaination\r\n

Headers_key1: header_value1\r\n

Headers_key2: header_value2\r\n

Webpagecontent\r\n\r\n ← two subsequent \r\n indicate end of response

Introduction to Networks (Protocol) contd..

Some common Response codes are -

200 - OK ← everything went fine

404 - OK ← The requested path was not found

400 - BAD REQUEST ← The request was malformed

500 - INTERNAL SERVER ERROR ← Something went wrong in server

Full Response codes and their explanations are available at-

https://developer.mozilla.org/en-US/docs/Web/HTTP/Status