Python socket network programming

https://pythontips.com/2013/08/06/python-socket-network-programming/

Hi there fellows. In this post I am going to take you on an adventure with python sockets. They are the real backbones behind web browsing. In simpler terms there is a server and a client. We will deal with the client first. So lets first begin by importing the socket library and making a simple socket.

```
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

Here we made a socket instance and passed it two parameters. The first parameter is AF_INET and the second one is SOCK_STREAM. AF_INET refers to the address family ipv4. ipv6 requires something different but we won't be focusing on it. Secondly the SOCK_STREAM means connection oriented TCP protocol. Now we will connect to a server using this socket.

I.Connecting to a server:

Well firstly let me tell you that if any error occurs during the creation of a socket then a socket.error is thrown and secondly we can only connect to a server by knowing it's ip. You can find the ip of the server by doing this in the command prompt or the terminal:

```
$ ping www.google.com
```

You can also find the ip using python:

```
import socket

ip = socket.gethostbyname('www.google.com')
print ip
```

So lets begin with writing our script. In this script we will connect with google. Here's the code for it:

```
import socket # for socket
import sys

try:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    print "Socket successfully created"
except socket.error as err:
    print "socket creation failed with error %s" %(err)

# default port for socket
port = 80

try:
    host_ip = socket.gethostbyname('www.google.com')
except socket.gaierror:
    # this means could not resolve the host
```

```
    print "there was an error resolving the host"
    sys.exit()

# connecting to the server
s.connect((host_ip,port))

print "the socket has successfully connected to google \
on port == %s" %(host_ip)
```

Now save this script and then run it. You will get something like this in the terminal:

```
Socket successfully created
the socket has successfully connected to google
on port == 173.194.40.19
```

So what did we do here ? First of all we made a socket. Then we resolved google's ip and lastly we connected to google. This was not useful for us though so lets move on. So now we need to know how can we send some data through a socket. For sending data the socket library has a `sendall` function. This function allows you to send data to a server to which the socket is connected and server can also send data to the client using this function. Now lets make a simple server-client program to see all of this in action and hopefully it will make your concepts more clear.

II.Making a server :

First of all let me explain a little bit about a server. A server has a `bind() method` which binds it to a specific ip and port so that it can listen to incoming requests on that ip and port. Next a server has a `listen()` method which puts the server into listen mode. This allows the server to listen to incoming connections. And lastly a server has an `accept()` and `close()` method. The accept method initiates a connection with the client and the close method closes the connection with the client. Now lets begin making our simple server:

```
# first of all import the socket library
import socket

# next create a socket object
s = socket.socket()
print "Socket successfully created"

# reserve a port on your computer in our
# case it is 12345 but it can be anything
port = 12345

# Next bind to the port
# we have not typed any ip in the ip field
# instead we have inputted an empty string
# this makes the server listen to requests
# coming from other computers on the network
s.bind(('', port))
print "socket binded to %s" %(port)

# put the socket into listening mode
s.listen(5)
```

```
print "socket is listening"

# a forever loop until we interrupt it or
# an error occurs
while True:
   # Establish connection with client.
   c, addr = s.accept()
   print 'Got connection from', addr

   # send a thank you message to the client.
   c.send('Thank you for connecting')
   # Close the connection with the client
   c.close()
```

So what are we doing here ? First of all we import socket which is necessary. Then we made a socket object and reserved a port on our pc. After that we binded our server to the specified port. Passing an empty string means that the server can listen to incoming connections from other computers as well. If we would have passed `127.0.0.1` then it would have listened to only those calls made within the local computer. After that we put the server into listen mode. What does `5` mean ? It means that 5 connections are kept waiting if the server is busy and if a 6th socket trys to connect then the connection is refused. Lastly we make a while loop and start to accept all incoming connections and close those connections after a thank you message to all connected sockets. Now we have to program a client.

III. Making a client :

Now we need something with which a server can interact. We could tenet to the server like this just to know that our server is working. Type these commands in the terminal:

```
# start the server
$ python server.py

# keep the above terminal open
# now open another terminal and type:
$ telnet localhost 12345
```

After typing those commands you will get the following output in your terminal:

```
# in the server.py terminal you will see
# this output:
Socket successfully created
socket binded to 12345
socket is listening
Got connection from ('127.0.0.1', 52617)

# In the telnet terminal you will get this:
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Thank you for connectingConnection closed by foreign host.
```

This output shows that our server is working. Now lets make our client:

```
# Import socket module
import socket

# Create a socket object
s = socket.socket()

# Define the port on which you want to connect
port = 12345

# connect to the server on local computer
s.connect(('127.0.0.1', port))

# receive data from the server
print s.recv(1024)
# close the connection
s.close()
```

The above script is self explanatory but still let me explain to the beginners. First of all we make a socket object. Then we connect to localhost on port 12345 (the port on which our server runs) and lastly we receive data from the server and close the connection. Was that difficult ? I hope not. Now save this file as client.py and run it from the terminal after starting the server script:

```
# start the server:
$ python server.py
Socket successfully created
socket binded to 12345
socket is listening
Got connection from ('127.0.0.1', 52617)

$ python client.py
Thank you for connecting
```

I hope this intro to sockets in python was digestible for beginners. In my server and client script I have not included exception handling as it would have boggled the beginners. Now I hope that you have a solid understanding about the working of sockets in python. However there's a lot more to it. For further study i recommend the Official python docs and this article on binary tides. If you liked this post then don't forget to share it on facebook, tweet it on twitter and follow our blog.

You might also like:
*) *args and **kwargs in python explained
*) Making a url shortener in python
*) 10 inspirations for your next python project