Raspberry Pi – Write your own network server!

A ***network*** is a collection of two or more computers connected together, either via wires or via wireless links. With a network you can send data from one computer to another. Programs on each computer make ***connections*** to each other, to make something useful happen. The ***Internet*** is an example of ***a Wide Area Network***.

Application programs on computers have to be run in order to take advantage of the network – a ***server*** listens for incoming connections, and a ***client*** makes outgoing connections. When a client connects to a server, the server ***accepts*** the connection and performs some service. The ***World Wide Web*** is an example of using a network application – the client is the ***Web Browser*** (e.g. chrome) and the server is the ***Web Server*** (e.g. apache)

IMPORTANT

In order to allow computers to send messages to each other reliably, the network must be properly configured. Usually this involves connecting each computer into a bigger network (or directly to each other) and configuring the ***IP Address*** and other parameters correctly both in the computer configuration, and in the application.

A Simple Server

Type in the following program using the IDLE 3 editor, and save it as `server.py`

Make sure that you get the indents correct, as they are important to the meaning of the program.

```
import network
import time

def heard(phrase):
  print(phrase)

network.wait(whenHearCall=heard)
print("connected")

while network.isConnected():
  print("waiting")
  time.sleep(4)

print("disconnected")
```

Testing your Program

First we will test this server just on your own Raspberry Pi.

A useful diagnostic tool we have pre-installed for you is “telnet”, which sort of means “make a telephone call over the network”. It allows us to test our server.

Open a lxterminal window and run your server:

```
python3 server.py
```

Open another terminal window and use telnet to test your server

```
telnet localhost 8888
```

We use “localhost” as the address which means “this computer”. We use port 8888 (which is the default port that our network.py module uses if you don’t tell it otherwise)

You should now be able to type characters into your telnet window, and when you press return they will be displayed in the server window.

How it Works

The `import network` line in the above program loads some library code (in this case, it is stored in the `network.py` file provided). This does a lot of the hard work for you in terms of setting up and opening network sockets so that you don’t have to bother.

`network.wait()` starts a server, and when it hears an incoming connection, it answers it and all messages get passed to the `heard()` function.

telnet is a standard diagnostics tool we can use to connect to any network service, and it allows us to connect to an address and a port,and then send it lines of text.

Note: If you try this workshop using an SDCard other than the one provided, you might have to do this first while connected to the internet:

```
sudo apt-get install telnet
```

That’s it! You have now written a network server using the Raspberry Pi!