Raspberry Pi – Write your own network chat

A network chat program is quite easy to write if you already know how to write a network server and a network client – there's not much more code that you need.

A Network Chat Program

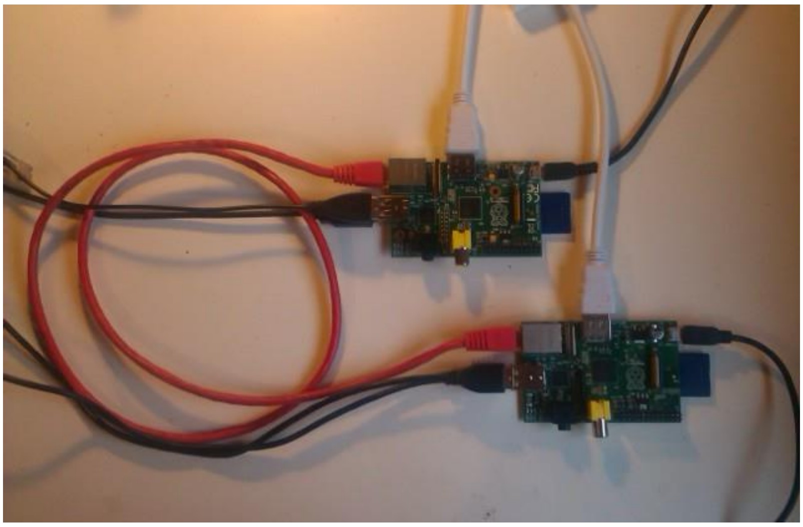
Create a new file in IDLE3 called "chat.py". It needs to look like the program below.

You can probably copy/paste most of bits of code from your client.py and server.py programs to speed up writing this program.

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

```
import network
import sys

def heard(phrase):
  print("them:" + phrase)

print("Chat Program")

if (len(sys.argv) >= 2):
  network.call(sys.argv[1], whenHearCall=heard)
else:
  network.wait(whenHearCall=heard)

print("Connected")
while network.isConnected():
  #phrase = raw_input() # python2
  phrase = input() # python3
  print("me:" + phrase)
  network.say(phrase)
```

Connecting Up



Testing your Program

You need two different computers to test this program – one is the server (accepts incoming connections) and one is the client (makes outgoing connections). Just think of it a bit like a telephone – one person waits by the phone waiting for it to ring, the other person dials the number. When the dialed phone rings, the person at the other end picks it up and you can now both talk to each other at the same time.

You will need to know the IP address of the server computer – see the networking worksheet where you learnt how to do this. (hint: `ifconfig`)

To run the server, from the command prompt, type this:

```
python3 chat.py
```

To run the client, first find out the IP address of the computer you are connecting to. Then type this at the command prompt (make sure you use the IP address of the other computer here)

```
python3 chat.py 192.168.1.2
```

You can now both type messages to each other and chat across the network!

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.

`network.call()` starts the client which connects to another computer via it's IP address. When data arrives from the server it calls the `heard()` function.

`network.say()` sends a message to the other computer you are connected to.

That's it! You can now send messages across the network with your Raspberry Pi!