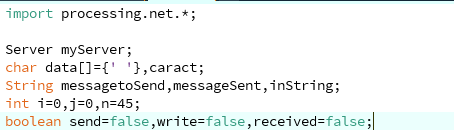
***ChatBox Documentation***

# Introduction:

* Processing was made for the purpose of being used by unexperienced programmers like I was and those that worked in the domain of visual arts.It has infinite uses if you are perseverent enough. At that time, in 2017-2018, I was interested in communication and networking, from this came the idea of a chat application. After I became confident with the basics of this language I wanted a challenge, so this is why I made ChatBox.
* At the time I did this project I could not find another application made in Processing that was about communication. So I started digging and found the “Network” library on their website (<https://processing.org/reference/libraries/net/index.html>). This library makes possible the exchange of data between different machines, allows you to create servers and clients which send, read and manipulates data sitting on a port made by you.
* Without any existing examples I had to think about the user interface and what goes behind the scenes.

# Server:

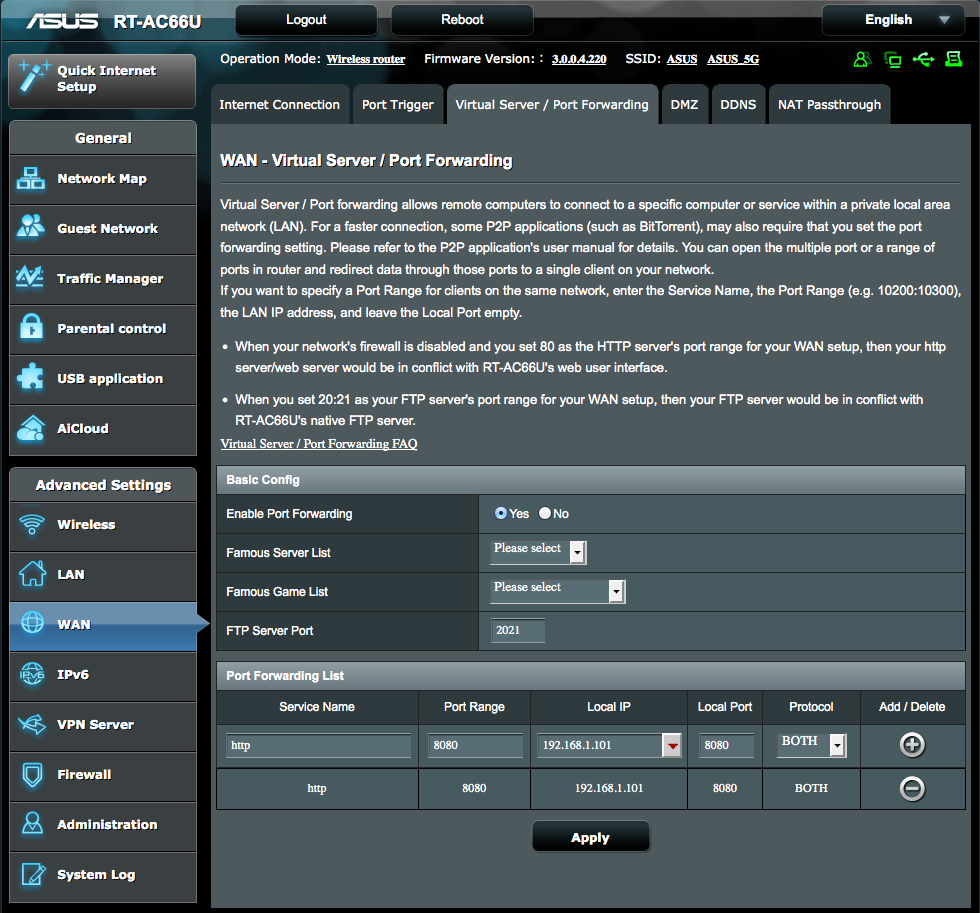
* The main components in Processing are void setup() and void draw(). In the setup() method you write lines of code which are run only once, sequentialy such as the dimension of the window, the background, etc.. And in the draw() method, the lines of codes are run continuously.
* First step is to import the Network library and set up some global variables for later use.



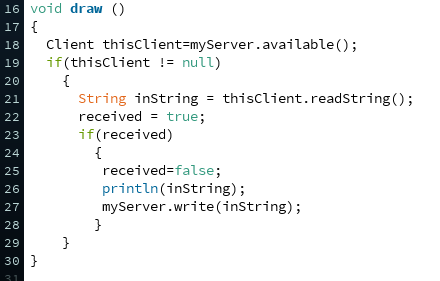
* In the setup() method we will introduce the dimension of the window and assign a random color, those does not matter as the server is not visible to the end user
* The parameters of the server object we make are as follows “this”, refers to the whole class, the port we opened on our personal router and the IPv4 of our machine.



* To open a port on our router we go to the settings of the router, we search for WAN and we do port forwarding.(for details on how to access the router settings search your router brand and model on the internet and read their instructions).



* And to find the IPv4 we run ipconfing in cmd on Windows, or ifconfig in terminal on Linux machines.
* No we go on to the draw() function, as mentioned earlier this part is run continuously. In this function we will see if a client has connected to our server and we manipulate the messages received to be send to the other clients.



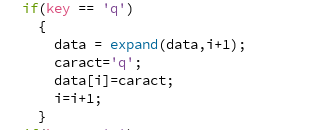
* On line 16 the available() function from the Network library searches for new clients, The client needs to be connected to the same port and ip address as the server was initialized. The server works as a simple bridge between users, each user sends to the server the message as a string, which includes its name as well and the server sends that message to every user.

# Client:

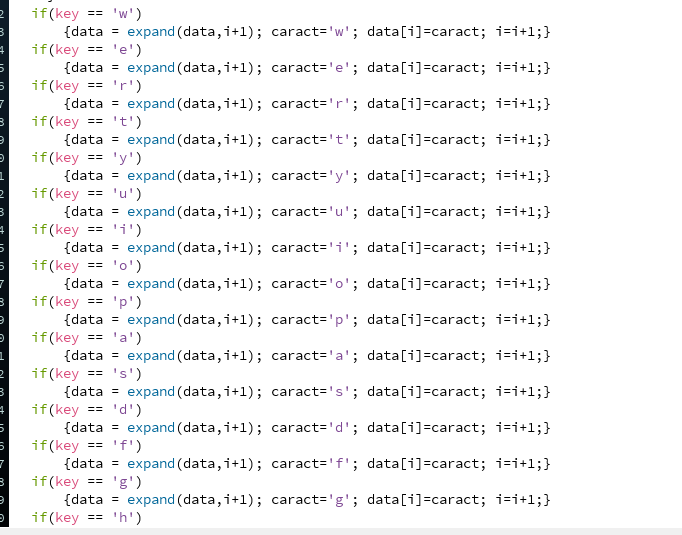
* First part of a client is the user interface.



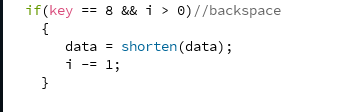
* The interface is made in the setup() function, because we will not need to change it frequently.
* As well, in setup() we make the Client object which holds the Ipv4 and port of the server the client needs to connect to.
* To make each client distinguishable, the client need to first choose a username.
* At the time this project was made, there was no usable keyboard function which displays what you write on the screen and keeps that in a variable. The software can only recognize the ASCII code of the key pressed during the run.
* An example of what happens when a key is pressed:



* For storing the key presses we use a vector which we named data, which stores the last key pressed and after that we expand the vector by 1 to make place to the new letter pressed.
* The “keyboard” object consist of iterations of the same code.



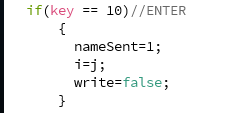
* For the commodity of deleting the last key pressed when we use the backspace is implemented as such:



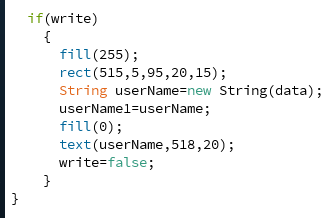
* The application has two states, the first is the one in each the client can not do anything if he does not chose an username, and the second one in which he can exchange messages with other users. For simplicity, we have 4 classes: draw1, draw2, keyboard, keyboard\_1.

## State 1:

* First state is the one where the client has no name yet. So the functions of the programme are different, in this state, no information is sent to the server and the name is stored locally.
* In this state the Enter key changes the boolean variable nameSent from false to true, clears the vector that stores key presses and makes the name field unwritable.

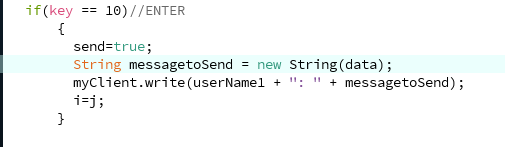


* The class which draw1, which holds the function with the same name, shows the keys pressed and stores the name in userName1. The way the characters are shown one after the other is by clearing the old contents in the field and replacing it with the new character array.

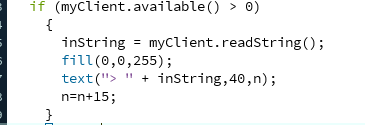


## State 2:

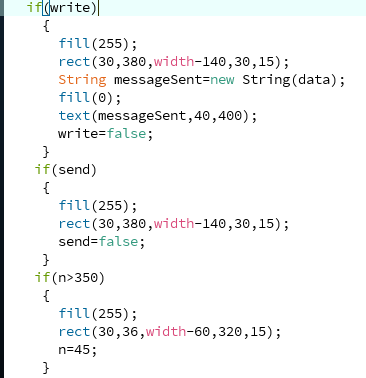
* In the second state, the username is chosen and we can use the application to exchange messages.
* This time when the Enter key is pressed, the client sends the message written to the server which then sends it back to all the users.



* The function draw2() verifies if the client receives something from the server and if it does it displays the data received in the chat field.



* This function also verifies if there were any new characters and displays the new characters written in the message field. If a message was sent, the message field gets cleared, if there is no more space in the chat field for other messages to be received, then the chat field gets cleared.



* To use those two states we use switch statements which changes according to the nameSent boolean variable.

