NTU Chat Server - Documentation

Benjamin Colligan

December 2020

Contents

1 A brief description of the programs aims **1**
- 1.1 An interactive user interface . . . 2
- 1.2 A means to enable communication between NTU students . . . 2
- 1.3 Simple accessibility features like 'tabbing' and high-contrast colours 2
- 1.4 Operation on a local machine between different instances of the software . . . 3
- 1.5 The ability to have separate chat rooms with separate user bases 3
- 1.6 Easy installation process . . . 3

2 Analysis of the programs requirements **3**
- 2.1 Programmed in Python? . . . 4
- 2.2 Shows use of previously learned topics? . . . 4
- 2.3 Sufficient complexity? . . . 4

3 Design **4**

4 Testing **4**
- 4.1 Can you send a message to the chat box? . . . 4
- 4.2 Can several messages from separate users be displayed in the chat box? . . . 5
- 4.3 Does the chat box update constantly after a certain period? . . . 6
- 4.4 Can the user set their own username? . . . 6
- 4.5 Can the user join a server? . . . 6
- 4.6 Can the users make their own server? . . . 7

5 Critique **7**

1 A brief description of the programs aims

Whilst developing the idea and concept of this piece of software I established mentally that there will be a short and simple list of aims at the end of the first phase of development. These are as follows;

- An interactive user interface
- A means to enable communication between NTU students
- Simple accessibility features like 'tabbing' and high-contrast colours
- Operation on a local machine between different instances of the software
- The ability to have separate chat rooms with separate user bases
- Easy installation process

1.1 An interactive user interface

I am for this piece of software to introduce a bit of feedback to the user when pressing buttons etc. this functionality is built into the Tkinter buttons a bit in the case of once you press down a button in inverted the shadow upon the button looking like it has been pressed in - therefore cementing the fact that the button has been pressed and to avoid any confusion.

1.2 A means to enable communication between NTU students

This is some what the overarching aim of the project and is the main aim I am focusing on if the other aims are deemed to be not possible in this development cycle or time is to stretched. What I mean by this aim is that I want a piece of software that can only be accessed and downloaded by NTU students that will provide an easy way to communicate between the rest of the NTU community - in the future I would love to implement users categories and a user search feature in order to allow users to find others that have the same interests and study the same subject as they do.

1.3 Simple accessibility features like 'tabbing' and high-contrast colours

Accessibility over the past couple of years in my education has become a big thinking point around all my projects, both in IT in college and now computer science in university. It's a necessity in both my past web development projects and software development projects. In order to tackle this issue within this project I have chosen a specific colour scheme that would strike a good balance between aesthetically pleasuring whilst also being easy to see for people with potential sight difficulties, as well as this Tkinter provides an automatic button tabbing function which makes it so within my program you can press the Tab key on the keyboard and which through all buttons and entry fields on the current scene.

1.4 Operation on a local machine between different instances of the software

Initially the idea regarding this aim was to make it be able to operate on all machines through a LAN network although that would include to developing and producing of a system which acts as a file server where requests can be made to change the contents of said files to further be represented on client piece of software and on the machines running it - in short the time frame and budget allocated could not fit that in. I instead decided to make it an initial milestone and a proof of concept to solidify the operation of the program which can be converted onto a LAN operating system later down the line.

1.5 The ability to have separate chat rooms with separate user bases

The aim with this is to allow users to not all have to speak in the same chat room. I have completed this aim by developing a way for the player to input their desired server ID, and if that server ID does not exist then to make a new server with that. Both giving the user the ability to create custom servers to their liking and also join their friends server with said server ID. Future implementation ideas plan for a user list for each server when joining in order to let all users know who can currently see that chat they are sending.

1.6 Easy installation process

Initially I intended to develop an installer which downloads the python scripts, and all the other files you need onto your computer for you, although since both the installation process is not difficult if done either manually or automatically and as well that there is currently no file hosting server in place to download the files from during the installation process we can make the installation process even easier by making the .zip file available to download and all that would need to be done is to have that .zip file extracted in a user-made folder for the program and then the program should operate as intended as long as the sub-folders inside the .zip stay original.

2 Analysis of the programs requirements

Program requirements in my eye are points of a program which is required as minimum in order to be classed as what it states it is - in this case 'a program'.

- Programmed in Python?
- Shows use of previously learned topics within the module?
- Sufficient complexity?

2.1 Programmed in Python?

This program is 100 percent programmed in Python, there are future ideas to implement a link to a HTML website in order to provide an operations manual although this is currently not implemented. I have also used modules like Tkinter and time to make use of Pythons extended functionality out of the standard install.

2.2 Shows use of previously learned topics?

This programs show use of previous learned techniques like functions, statements and data structures etc. as well as advanced, less-covered, programming techniques like file handling in order to keep track of messages sent in chat room. The use of Tkinter could also show proof me as a developer delving deeper into Python and it's abilities in order to complete these programs aims stated earlier in this report. I also experimenting with the threading module in Python which allows for mutlithreading capability in Python in order to run a function outside of the Tkinter mainloop() that makes it so only GUI code can be executed and any code outside the mainloop() would be executed after the GUI closed. This problem was eventually fixed by using the universal widget after().

2.3 Sufficient complexity?

This some what ties in within the previous section and can be backed up in a similar manner. As well as using all the techniques previously taught to us, this program for-go's using simple statements too much and tends to lean towards to more efficient but more difficult approach available to achieve each goal. For example, instead of storing the current server ID, the current message written in the message entry box etc. I have opted to use the get() function for each widget I need the information for whilst running the updateChatBoxText() function in order to maintain quality and integrity of the information which further more makes sure that I am updating the log with up-to-date information.

3 Design

The design for the this program is very simple and intuitive as per my program aims. Below are several flow charts dictating the process that the program goes through when the user activates the next scenario.

4 Testing

4.1 Can you send a message to the chat box?

- Expected outcome: Typing a message in the message entry box and clicking send makes the message appear in the chat box

Figure 1: The steps included in the updateChatBoxText function()

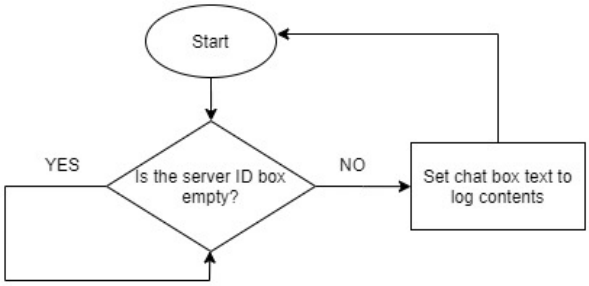
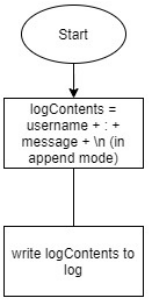


Figure 2: The steps included in the updateLog function

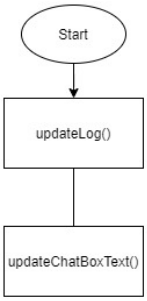


- Testing method: I typed 'Test' into the message entry box and clicked send
- Actual outcome: "Ben: Test" was printed into the chat box

4.2 Can several messages from separate users be displayed in the chat box?

- Expected outcome: Opening a new software instance and typing in a new message will add onto the previous
- Testing method: Opened a new software instance, changed username to Ben2 and typed 'Test'
- Actual outcome: "Ben2: Test" was printed into the chat box below "Ben: Test"
- Further action needed?: No

Figure 3: The steps included in the sendMessage function



4.3 Does the chat box update constantly after a certain period?

- Expected outcome: Opening a new software instance and typing in a new message will add onto the previous
- Testing method: Opened a new software instance, changed username to Ben2 and typed 'Test'
- Actual outcome: "Ben2: Test" was printed into the chat box below "Ben: Test"
- Further action needed?: No

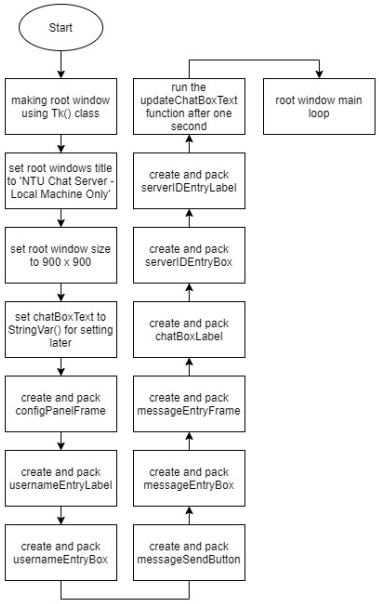
4.4 Can the user set their own username?

- Expected outcome: The text in the username box is reflected in a sent message
- Testing method: Sent a message of 'Message' with 'Bob' entered in the username field
- Actual outcome: "Bob: Message" was printed into the chat box
- Further action needed?: No

4.5 Can the user join a server?

- Expected outcome: The username is loading off the correct server message log file
- Testing method: Sent a message of 'Test' whilst '123' is in the server ID field
- Actual outcome: "Ben: Test" was printed into the chat box and 123.txt updated to read "Ben: Test"
- Further action needed?: No

Figure 4: GUI configuration and building



4.6 Can the users make their own server?

- Expected outcome: If the user enters a non-exist server ID a new server is made
- Testing method: Deleted all logs, except 123.txt and then entered 1234 into the server ID field
- Actual outcome: Upon sending a new message, 1234.txt is made within the /servers directory
- Further action needed?: No

5 Critique

There are most definitely a couple of things I would critique about this piece of software regarding what I in-visioned it as in the beginning and what it is in this current moment in time. The most obvious different is the lack of LAN support, although this was only a time-frame and budget issue I think it's definitely in reach now that I have the core operations of the program completely smoothly and all I would need to complete is a bit of further research into Python networking and then I could possibly rather easily convert the code over to work in a LAN server setting between different PC's. As well as this my other glaring critique with this software is the practically of the security when in a real life use

case, due to time-frame issues I am storing all the would-be server files locally with the source files themselves and of course if that were to continue into the release of the software then troubles rising to out of date server lists and server mismatches would become incredibly evident very quickly - so in the future I shall host a file server on my own machine in order to store and sync all the server-side files my program needs for operation.