Development Journal for My 20 Percent Project

Noah Guidara

Week 1 (Friday, February 05, 2016):

Today I started out my 20% project by watching a YouTube video on the basics of creating a Twitch.tv chat bot, the logistics of actually seeing results of which will be difficult, as the school internet blocks the website. However, I can test at home with my own channel (which I will be omitting for the sake of privacy), and I can still develop in class on Fridays.

The fundamentals of the bot have been laid out mostly, as getting it to barely function is quite a menial task. All this requires is a TCP client connection and some other basic methods to connect to the server. In the upcoming Fridays, I’m going to tackle the vocabulary and inner workings of the bot code, as to understand what I am actually doing; copying doesn’t mean anything if you don’t know what it means.

My hopes for the project so far: Pretty high. The challenging part will be to debug and hard code in exceptions to everything, as the end user seems to break anything they touch. The end goal of this project is to get a bot that makes a channel chat better, and not to be there for the sake of having a bot.

Total Word Count: 206

Week 2 (Sunday, February 14, 2016):

Over the weekend I realized that I was doing some things completely wrong. So I re-watched what I did over on the 5th and figured out what I was doing wrong. Windows Form Applications have nifty toolbox items that allows someone to display or count time. Working from home also allowed me to test the simple ability to read and display what is going on in the chat (See Images Folder), and to my enjoyment, it worked.

My end goal is to understand the code, since understanding what you make is a good way to improve on what you make. Getting the functionality I want in this bot will take time and effort, which whether I want to or not, is essential to getting an awesome bot (and a good grade).

I will continue to cross-reference tutorials and other bots to see what functions they serve. A notable one, NightBot, can do many things. It can act like an automatic moderator, it can have keywords or commands and act accordingly, and it can run with other plugins to help the streamer.

Total Word Count: 181

Week 3 (Sunday February 21, 2016):

I have made a lot of progress. The bot actually does stuff, and it does it very well! It can respond to input, can read the user that said it, and can even do mod actions if the channel owner so pleases. This is a major update and I will continue to make it better to actually transform it into a bot that could be used by other channels.

That being said, I fear for my deliverable schedule. I think I have made so much progress, or I massively underestimated the ability to create the functions, that I have basically met all deliverables by the third week. Such being said, I will always strive to improve upon it, and also make it much more user friendly. At the moment, it’s a bot only a developer could love.

My current goal for the next week is to reformat everything. The source code as I’m writing this is horrible, redundant and a pain to read. Since this is an open source project, the methods will have to be public, and I will have to create an API sheet and document all methods and features. I am very glad I took this project; it has taught me a lot. Hopefully, by the end of the development time, I will have a bot to be proud of.

Total Word Count: 223

Week 4 (Sunday February 29, 2016):

It was only a minor update, but I added a log feature. The log feature creates a .txt file that each channel gets and every time a user speaks inside the channel it logs the time, the message, and the user. It also automatically generates dates and everything is managed via the C# code.

Since most of my deliverables are already met, I’m adding a stretch goal. By the end of the weeks in May, I will try to set up a server and a GUI to manage everything. My plan is to actually have a legitimate contender for good twitch bots that large channels will use. The server will host any assets such as condition files, log files, and other information files that I might add on later.

It’s been quite an experience working on this project, but I hope by the end of it I come out with much knowledge to help me in the realm of computer application design.

Total Word Count: 162

Week 5 (Sunday March 6, 2016):

I didn’t really do anything this week. With a TSA (Technology Student Association) competition coming up and a game to complete, full focus has been put into that. Besides looking into a server to store information and creating a user interface, my bot is looking presentable (although the code could use some shaping up).

Working on this bot and other projects relating to computer science have shown me that hard work and progress take time, and that everything is hard until you start getting into it. The game I’m developing and this Twitch bot have also been applicable, so I feel better about working on them. It’s been a wonderful experience and I will continue working on stuff in the future like the sort. As for now, I’m tired and stressed, and I don’t want to destroy my physical and mental health.

The bot can wait, this project cannot. I’m leaving March 10th for Twin Falls, and there I will learn much about engineering and what it can do for me. I’m hoping that all this hard work was worth it, that I’ll get an end result from all the madness. Applying computer science concepts has been great, and I will seek to continue it in the future.

Total Word Count: 154

Week 6 (Sunday March 13, 2016):

So I’m back from my competition in Twin Falls. It was pretty neat and I got to see and go to many places. As for my competition, I finished second place, so not too badly. As this comes and goes my stress also leaves. Schoolwork should be the only focus for me now, and that for the most part is manageable.

Next week I’ll be working on the bot to reformat and improve it. Adding additional methods and processes to make it better than before will come. The whole video game thing has sort of turned me away from programming for a while, and I hope this isn’t permanent. I tried testing it a while back while I was focusing on TSA and it didn’t seem to run for some reason, apparently it needed a file. I’m thinking about moving this onto another project and redoing everything from scratch with reference to the first draft. It’s a bloated assortment and it needs to be contained where it can be.

Getting back into schoolwork and this project will be a tough transition, but I’ll tough it out and manage to get an end result to be proud about. Homework will probably be a first as I need to catch up on that too.

Total Word Count: 212

Week 7 (Sunday March 20, 2016):

It was the beginning of spring break and I wasn’t really interested in working on the bot besides getting the framework set up. I started a new project file and got the basics of the bot working. The general idea of connecting through the TCP client was the same, but I went forward in a way that would allow me to add other features without destroying everything. Also, a big feature that this would allow me to have is to add commands. My old project was only one .cs file, while this one is much more spread out and allows for wrapper classes that store vital information.

I will start to work on the bot over the break a little, but I will have functionality by the end that is greater than my previous bot. I hope to add the ability to create and store commands for specific channels. Also I will use my previous experience to create other functionality my previous bot didn’t have. Largely my greatest issue with my bot has been getting the .txt files to not mess up, as they usually say they are in use when they aren’t.

Total Word Count: 193

Week 8 (Sunday March 27, 2016):

I made good progress on my bot, and now it has some new features. One that didn’t really require that much coding but a bit of design prowess that I hadn’t been utilizing before was the windows form and creating buttons and text boxes that would allow for a different channel owner that wouldn’t know how to edit the source code to do stuff. Mainly, it just allows for the bot to switch channel chat rooms and send messages via the bot, but it also allows the person to exit the correct way (or else data doesn’t get saved). One function that required a bit of savviness but in the end was a great idea was to add “!addcommand” which allows me, or the channel owner, to add their own commands to the bot. Currently there is no storage within the commands in .txt files but that can be easily added.

Upcoming features that I plan to add are the storage of commands and other stuff, and allowing the bot to recognize mod actions and commands. Even if everything doesn’t turn out perfect at the end of the day, I’m still very proud of what I have learned over this whole experience.

Total Word Count: 202

Week 9 (Sunday April 3, 2016):

I added some wrapper classes to help contain the mess that are my files for each channel. So far I haven’t implemented them, but I will be doing that shortly. For the next couple of weeks productivity will be low if I don’t do it on weekdays because the NA LCS playoffs are going on. Hopefully if there are no hard projects to work on and I am feeling well I will be able to finish the bot in its beta build and get something that is actually useful done. My hope now is to be able to implement some libraries to help me out and get some code exported over there because this is becoming a tree with many branches. In the next few weeks the work will just be adding some functionality and making everything look and feel good. Whether this will go smoothly is a different scenario.

This project has been pretty fun, and of all the channels I’ve tried it out on they seemed to be impressed with the results. I have come very far since the start of the year, and I hope to further continue my experience making other things in windows forms that are not bots.

Total Word Count: 203

Week 10 (April 10 2016):

Again, I am unsatisfied with my implementation of a successful bot build. This upcoming week I’m going to lay out a plan and figure out how to make a system that works efficiently and in a way that makes sense. My previous bot made features very hard to add, so this is why I’m starting on a version 3. Not in the sense that it’s 3 times better than the first one or another addition, rather a different way of going about the whole process.

Rather than becoming frustrated with my current version and trying and catching all the errors via hard manual testing, I’m going to create a solution that may finally solve all my problems. The idea is to go forth with a framework, not a build (like my last version), that can support different versions of my bot if needed. I will work on this over the week not by starting on Visual Studio, but on a design plan, that hopefully is the best way for me to understand how to do everything.

Total Word Count: 176

Week 11 (April 17, 2016):

I have created a base layout plan for my bot and have added some basic running features. Online is not yet added but once I get some of the groundwork laid out the rest should be easier than before. Talking on streams with experienced and professional streamers has helped me to figure out and test some of my ideas and problems. The following week I’m planning on separating my components using namespaces and DLL’s and creating a valid work log using Trello. I have learned that if you don’t keep track of what you did last time or if you don’t come in with a business plan, you won’t get the results you need or want and you just end up wasting your time somewhere else that doesn’t need the support.

Watching Ludum Dare has shown that incredible progress can be made in very few days and people made games that I wouldn’t be able to make in a couple months. The experience from this project will hopefully throw me into larger opportunities such as making another Unity game or making other software using Windows forms. For now, I’m smarting small, and making this bot no matter how much iteration it takes gives me the experience needed in the corporate world. Designing a layout before I make the project, as if I’m my own client; splitting everything up into manageable chunks; even managing my progress; all of these are part of the business process.

I’m still hoping I’ll have time to be able to implement my stretch goal, as I’ve receded a little after making that. Coder’s block is a very real thing for me and sometimes it’s more of a chore to make this bot than it is a hobby. I need to make a schedule for myself, and I need to take better care of myself so I can feel in the mood to make this bot. That’s all that I have planned, but by the final deliverable and the final due date, I will have a bot that expresses my progress from all the weeks.

Total Word Count: 348

Week 12 (April 24, 2016):

I started my final version of my bot. Worked on my DLL and created storage and timing inside of it. I have everything I need now to complete the project. The Windows Form application is as close as it gets to drag and drop, and managing tabs and text boxes is going to be extremely easy, although I wish it wasn’t as explicit.

Working with my own personal timer is going to be very irritating sometimes because Windows Forms doesn’t like it if you call other methods on other threads, as things can get messed up. This is where I learned that delegate methods and lambda expressions are things of beauty. It’s like having an interface that you can carry around and put in as a parameter on everything, and just change and execute it whenever you please. This has been extremely helpful especially with my file editor and GUI manager.

Once everything gets rolling and I get the I/O situation figured out, everything else should be a breeze. Once I learn and pick out a language to use for storing my commands (I’m thinking JSON is the preferable choice), storage should be a very straightforward task.

Going through the final iteration has made me look back and realize how inexperienced I was back then with C#, even with Java, and that the best way to learn is to practice and struggle on your own. Luckily I have derived methods in which I don’t get errors due to my experience with stressing out (I’m looking at you parameter requiring user getting channel owner) for hours and toiling over a little mistake. Things get done quickly and I feel it’s not just me getting through the mud, but rather building tracks and starting the train. With the deliverable due next week, I will have my interactive bot that everyone so very much expects from me (I’m looking at you Isaac).

Total Word Count: 319

Week 13 (May 1, 2016):

So the bot has about as many features as before, but the command memory for each channel now works and removing commands also works great. Some GUI issues need to be fixed but that will come next week. Now I’m implementing a user list that allows someone to check users and act accordingly. The project is nearing its end and it’s been quite a piece of work. I’m also going to work on creating my presentation and practicing it because I have a lot to talk about. Unfortunately I’m not sure getting a server up and running will happen, but I am going to figure out a way to compactly store all this stuff.

Here is a complete list of what I’m going to implement before the end of the project (Hopefully!):

1. Able to ban or restrict words (as in if it’s said too much ban the user who said it)
2. User list and features coming with it.
3. Time and uptime of the channel (not sure about the latter)
4. Information through commands
5. Finishing the GUI

Total Word Count: 180

Week 14 (May 8, 2016):

I didn’t do as much work as I should’ve over the weekend and in general, so I didn’t get the deliverable done. However I have implemented a couple more classes to deal with owner and mod only commands. Just need to get the storage working with those and I’ll be golden. After that is done there needs to be a massive GUI layout because it’s a mess at the moment and doesn’t really do much. After that, there needs to be testing done on everything, and it needs to have the same if not more features on it. I might scrap the word restriction but I can implement the base informational commands that just come with every channel such as “!credits” and stuff like that. Everything is a mess, and it isn’t nearly as straightforward as I thought this final build was going to be, but since I don’t feel like putting in the effort of streamlining it, I have to make do with my go to solutions.

Here is a list that I will 100% implement before presenting:

1. User list and features coming with it.
2. Time and uptime of the channel (not sure about the latter)
3. Information through commands
4. Finishing the GUI

Total Word Count: 207