Jacob Smith 5/16/2019 5:06 AM: I looked up how much information there is on Object oriented programming In Arduino, and <https://forum.arduino.cc/index.php?topic=6127.0>

Post 3 says that writing libraries is difficult, which helps to justify my automatic library creator

Jacob Smith 6:53 PM: I wrote a series of Java classes to allow for the user friendly creation of new Arduino Libraries. The ScriptEditor class allows for the simple reading and writing of files, the Arduino Class allows for the parsing and editing of Arduino Classes, and the GUI class creates a simple user interface.

**Graphical User Interface of Library Creator🡪**

I made these libraries because I found myself repeating a series of tasks every time I made a library like replacing the name and changing the header comment. The user interface accomplishes both of those actions by reading user input in textboxes and using it to modify a simply example class.

**Template Class for Arduino Code generation🡪**

While this isn’t that helpful on its own, these classes show the possibilities for making it easier to create arduino classes, in the future the user could add a global variable and have both the header and cpp files automatically updated, currently only the cpp file is changed.

Jacob Smith 5/13/2019 5:19 AM: I am rewriting the Java ArduinoClass to build a cpp file from a template, and I am researching arduino libraries some more. In addition to my current scheme, auomatic conversion of an arduino sketch (.ino) to an arduino library (.cpp) would be very useful

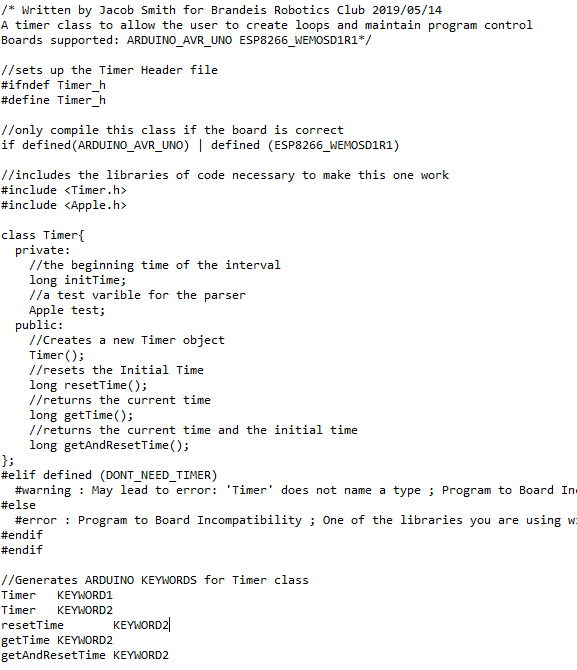
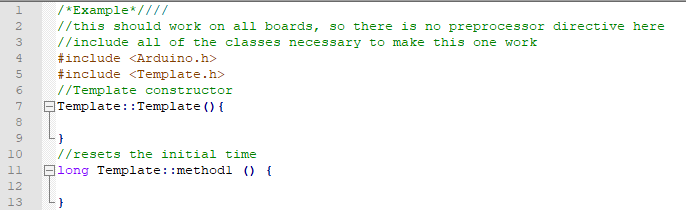
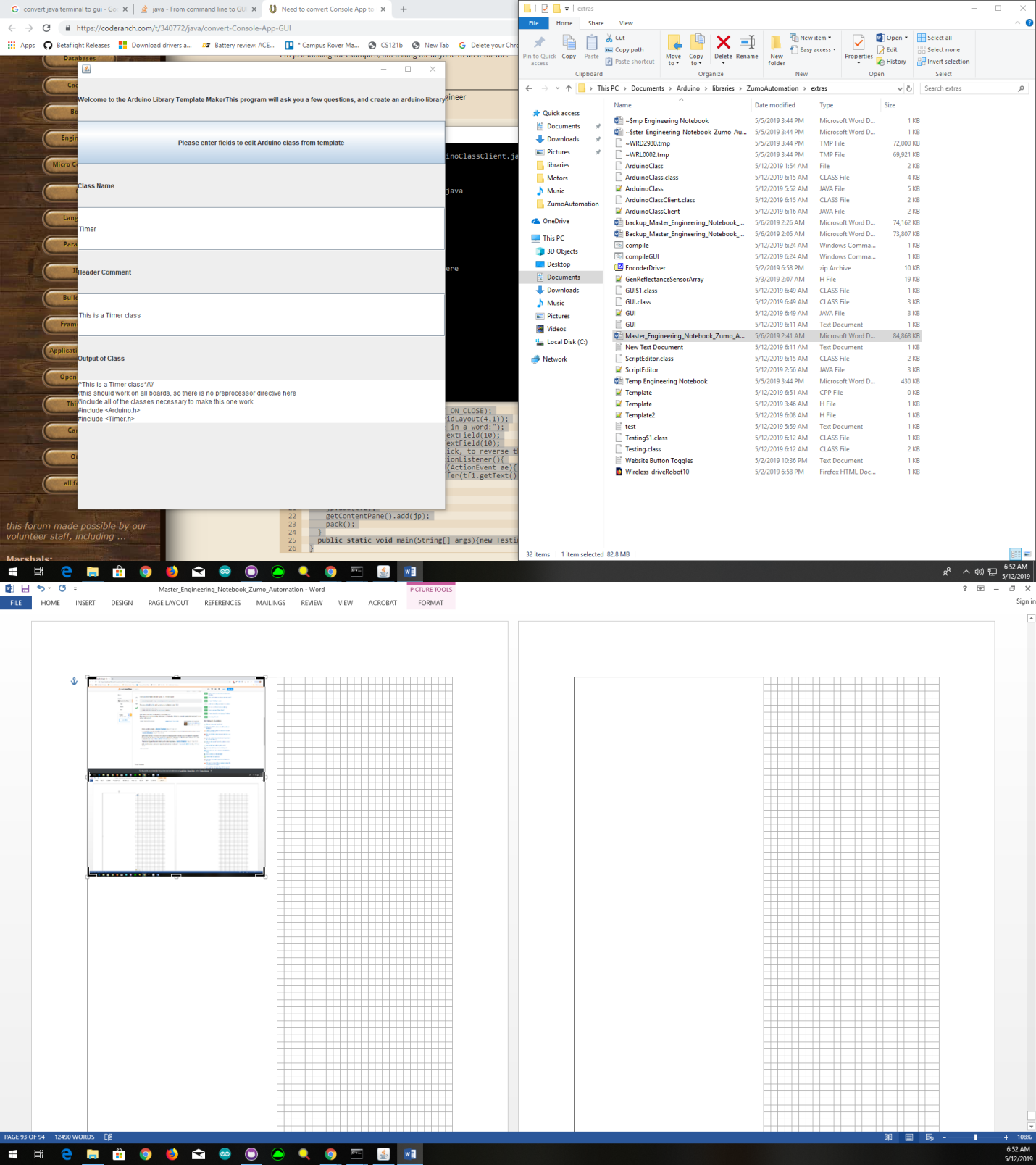
[1] <https://arduino.stackexchange.com/questions/32998/how-to-convert-arduino-example-sketch-to-a-complete-c-project>

[2] <https://community.platformio.org/t/tutorial-for-creating-multi-cpp-file-arduino-project/5830/13>

5/14/2019 Jacob Smith: I wrote an inheritance hierarchy of common methods in ArduinoClassMaster, body methods in ArduinoClassCpp, and header methods in ArduinoClassH. Left is Autogenerated header and keywords file from ArduinoClassH

**Automatically Generated Header file🡪**

Jaocb Smith5/15/2019



Jacob Smith 5/15/2019 1:23 AM: I restructured the github folders, here are some toggle button schemes that Deveroah and I thought of a while ago: **Wireless Button Driving Schemes🡪**

I have been working on the user interface for Arudino class generator, and I am fidnign hat putting the required varibales names, prompts, and example formatting in an enum makes my client code easy to read and write. 🡪

**Motion-Stop**

Dive F -stoppped

Drive B -stopped

Turn L-stepped

Turn R=stopped

**Direction-Turn**

Drive F-DriveB

Turn L-Turn R

Move- Stop

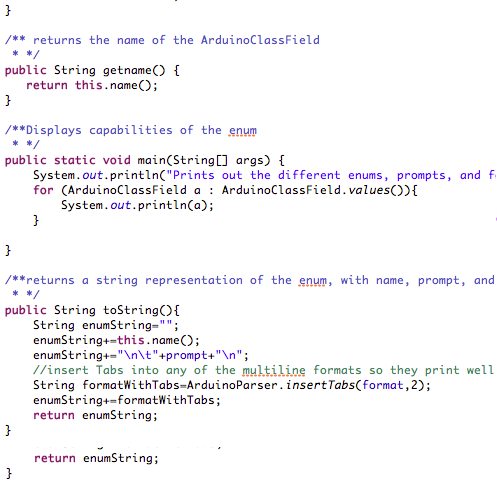
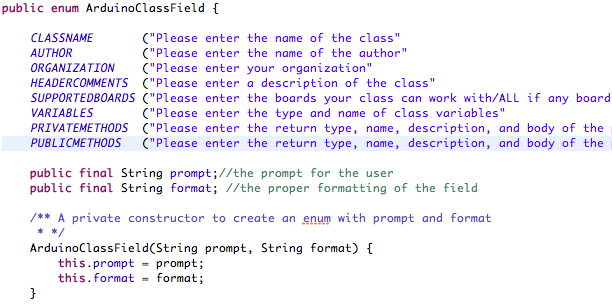
**Pin Directed**

Drive--Stop

Forward--Backward

Turn Right--Stop

Turn Left--Stop



**ArduinoFields Enum for Code Readability and to allow all field related prompts to happen In one place**

My Journal Notes on the Project

Unknown Date:

void|driveLeft|tells robot to Drive Left robot.driveLeft()

void|driveRight|tells robot to Drive Right robot.driveRight()

had to reset project tested prompt method in prompt enum, going to display DONE prompt more cleanly and also, I have to break up internally parsed prompts better I am using recursion to handle the complex formatting🡪

is on the innermost level tokens of |, then tokens of newlines, then tokens of double newlines while this format is complicated, it is both computer and human readable

**Example of Method Formatting**

I am working on a field by field prompter, handling initial conditions before loop does simplify code

Sunday: I am only allowing allowed characters, to hide which characters I am using for parsing and also exclude strange characters I haven't thought of

I am replacing all instances of scanner read next line with my validating looping method

It seems like using an enum for prompting is too complicated, I will just use methods

I am handling special cases manually, simplifies my methods and makes code more readable

I am using interface to save code enums can implement interfaces

Manually typing in user responses for testing is tedious, so with print streams and enums, I can now just run a test instead of manually typing in test input for ArduinoClassPrompts.

I am now at the point where I can prompt field by field and try to generate the Arduino class, but I found methods are missing | character set breakpoints as you work with debugger, maybe directly from runtime exception found a parsing error in variables, explains why only header file had it, I’ve got an extra newline in variables I fixed that error which was missing |, but methods had extra newline, I removed it with substrings Now test runs correctly, but public methods aren't showing up TO DO: not do substring by flipping with fencepost solution, fix public methods, unit test error corrections public methods was newline issue,

Now that I added constructor, the output essentially looks like an Arduino class. I switched order of private and public methods in body file, which puts constructor at top and puts most visible methods at top too line by line unit test passed.  
​  
Created Arduino class maker GitHub repository so I can actually delete unused code and know it is backed up

May 23 I am working on the program that converts an existing Arduino sketch to a library, which required a lot of background code work. I modified the MiniScanner class to display an error with the word that was looked for, and the Arduino Class generators to allow method parameters of null if blank. The methodParser class handles the work of converting a method into the format the code generator uses, and I want the program to automatically generate an example sketch. The sketch parser should also be able to tell which methods are public vs private by looking for which methods are in the setup and loop methods The parsedMethod class has fields even though I could have the whole thing be one big toString. The SketchParser class currently can read a sketch into the code representing the header, variables, public and private methods, loop and setup, but they aren't formatted into the format the Arduino class generator needs

Now that the SketchParser method can convert my sketch into a fairly well formatted list of header comment, variables and methods,the next step is separating private and public methods and parsing variables. The variables parsing will become fairly interesting/complicated, because they should all go to a constructor. This means that the sketch to library program is forcing me to add some features like adding constructor and adding method parameters.

The method correctly decided that wifiConnect and run server where public methods I am using replaceAll String method a lot, very useful in reformatting sketch. I am writing ParseVariable method with test cases to convert code of variable declaration to parsable format. This is complicated because data types and values can be more than one word, and arrays are in form type name [], and variables don't have to be initialized .

5/24 10:51 PM: Working on Arduino parser and unit tests, I got test to pass when variable is declared not initialized, still failing Linked List switch nodes (I wrote linked list to iterate over variables which can have multiple words like const char) and eliminating extra vertical bars. passed extra vertical bars test using String. It seems that using String.replace works when String.replaceAll does not, I don’t know why.

5/25 10:48: FUTURE WORK: I want to integrate this project into the wider Arduino Environment by reformatting and compiling files before they are parsed. This would allow me to standardize text before I have to parse it, and return an error in advance if program doesn't compile. Also, I want to eventually make a pull request to have the automatic class generator be part of the Arduino ide. Relevant links:

[1] Arduino Command Line Interface <https://github.com/arduino/Arduino/blob/master/build/shared/manpage.adoc>

[2] Jar files already part of Arduino <https://github.com/arduino/Arduino/tree/master/arduino-core/lib> [3] How to create a jar file in eclipse <http://www.skylit.com/javamethods/faqs/createjar.html>

May 27, 2019 10:34 AM: I looked up who else is working on an Arduino Class Generator on the Arduino developers’ group, no one there is. I'm working on passing more of the VariableParser unit tests, so I wrote a MiniScanner getRest method to help.

CODE STYLE: Else statements are useful in avoiding something right before exiting a loop. In that scheme, use Boolean loop control variable as condition. Set variable in loop, and use else statement to do whatever should be done while looping is chugging, but not when it is about to exit. This is a more subtle version of a break statement.

12:58 PM: Working on LinkedList switch method for variable parser, I wrote a state method that handles null pointer exceptions, and made setter methods private to force its use

1:13 PM: Arduino IDE Auto format inserts spaces but doesn't remove them, I'll have to do that on my own, which should be using the MiniScanner class internally.

3:07 PM May 28, 2019: I got the MiniScanner to allow it to ignore multiple tokens in input, but now a lot of other unit tests fail

6:58 PM: I spent a lot of time today trying to get rid of scanner errors, and put the project on GitHub so I wouldn't have to redo my work again

11:00 PM I got all unit tests to pass, and having project on GitHub with commits on passing tests makes it much easier to keep project moving forward with unit tests passing, and creating branches is useful. In the SketchParser class, I am working on an autogenerated constructor based on variables. Before that, I am preparing the program to handle the <https://www.arduino.cc/en/Hacking/LibraryTutorial> Morse code sketch as is so I can post it to the Arduino developers forum [https://groups.google.com/a/arduino.cc/forum/#!forum/developers](https://groups.google.com/a/arduino.cc/forum/" \l "!forum/developers). The challenges so far with that have been to reformat the newline brackets style of coding into the same line and to create to do comments when comments are missing, which is what I'm working on now.

May 29, 2019 10:37 AM: last night, I got the SketchParser class to produce keyword, example file, body, and header for the ESPServer and Morse classes. The unreliable parts are automatic constructor generation and how the sketch file doesn't call based on object. In addition, I posted my project idea on the GitHub developers forum, and I got a response from someone to share my GitHub link, although I haven’t decided whether to keep my code private yet.

7:18 PM: I got the generator to change method calls in example sketches to be an object method, like converting dash(); to morse.dash(). Also, I got the GUI to be useful, now the user can click on

RUNme.jar, and this window pops up🡪

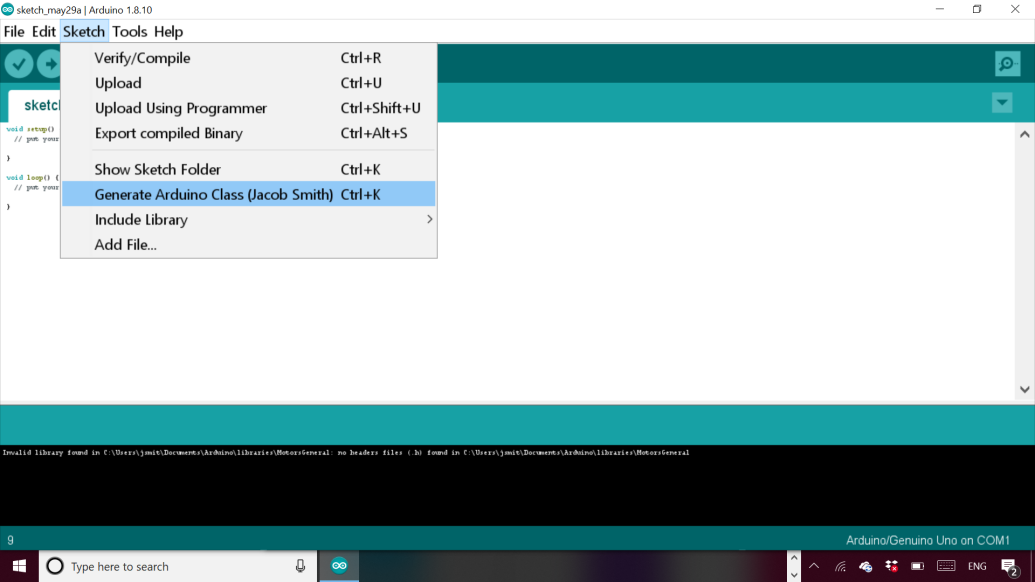
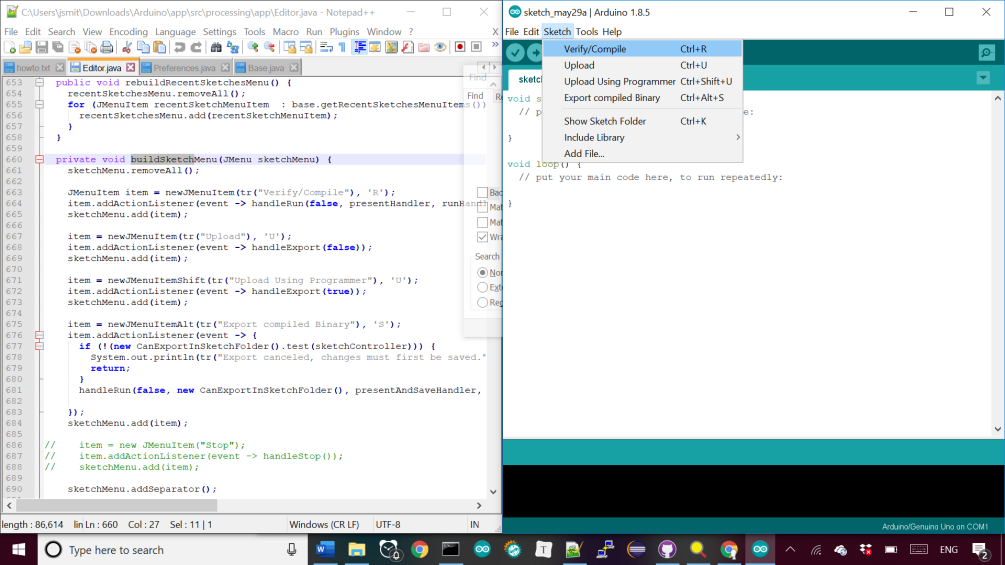
The user can enter the name of the class and location of the correct Arduino sketch.

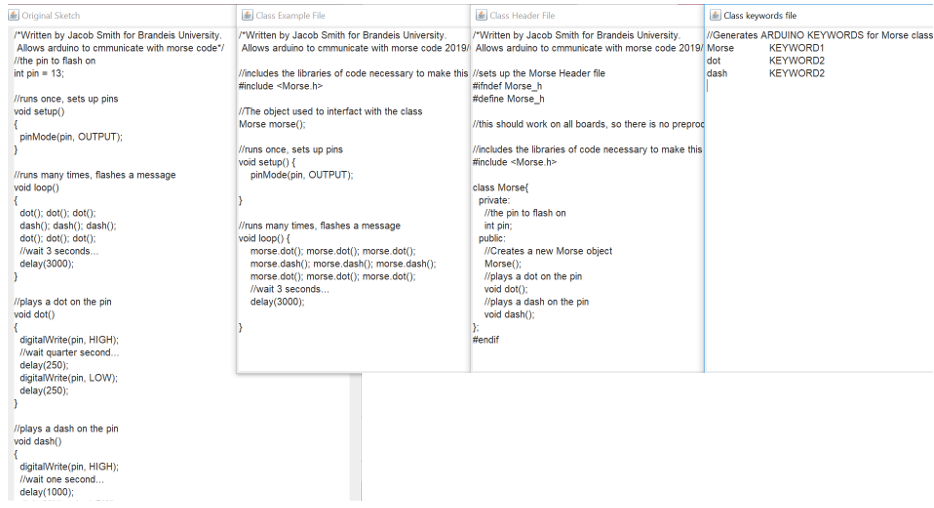
Then, when the button is pressed, the following windows pop up (next page).

Future work: Integrate this into the Arduino IDE: <https://github.com/arduino/Arduino/wiki/Building-Arduino>, research whether I should publish this program.



**User input interface, runMe.jar**





(TOP) Generated Example, Header, keywords, and original sketch. From runme.jar

(BOTTOM) Generated JDK error message because FJD must be 32 bit

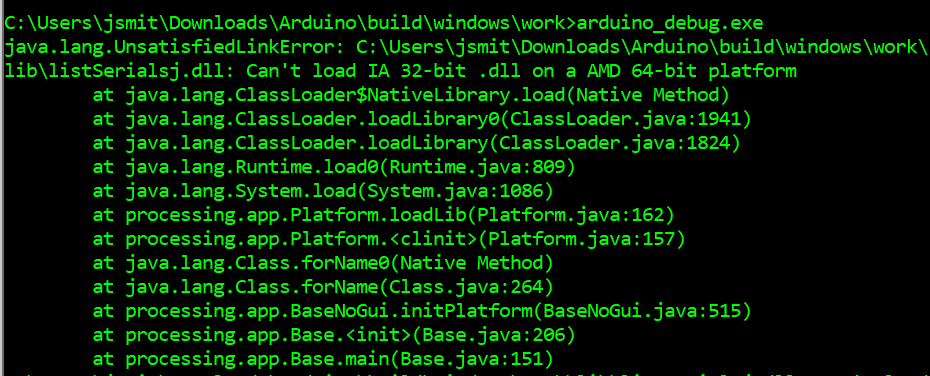
How to build Arduino IDE<https://www.mkyong.com/ant/how-to-install-apache-ant-on-windows/>

Why I need 32 bit JDK

<https://github.com/arduino/Arduino/issues/3276>

May 30 1:00 AM: I can now compile and run the Arduino application; everything is in the Editor class. I can now add a menu option called generate Library, and I’m trying to use the tab creation feature to generate all the correct files at once. Then, I need to add the relevant classes into the Arduino folders.

May 30 2:52 PM: I am trying to automatically create files in Arduino ide, relevant functions: Addfile Sketch.java line 311, SketchFile.java line 94, EditorTab.java createTextArea line 146, EditorTab.



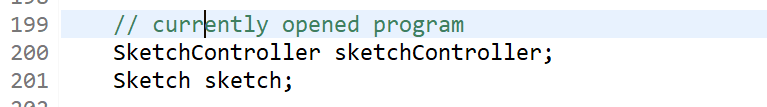
(Middle)🡪 The menu option to generate Arduino Class.

(Bottom)🡪 The area of the Editor class in the Arduino IDE that I will be modifying for the GUI.

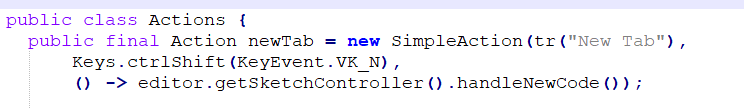
SketchControler.java 136, nameCode

Sketch Controller Line 598 import Library

Sketch Controller upload line 598 (some boards have a password)



**Editor.java line 199 fields, sketchController allows new tabs to be created**



**Editor Header.Java Line 88 , shows how new tabs are generated**

Relevant background:

<https://github.com/sudar/Arduino-Makefile> More control of Arduino Compiler

Arduino Tabs are just a way of breaking up a sketch <https://forum.arduino.cc/index.php?topic=206078.0>,

Nice tutorial for library creation <http://arduino.land/FAQ/content/7/43/en/breaking-a-sketch-into-multiple-files.html>

Using sketchContoller code avoids java exceptions and lets you create new tab.

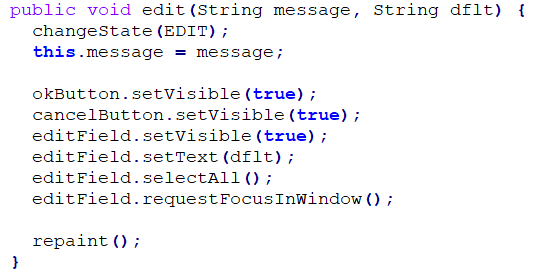
11:00 PM: I am writing a script to compile the Arduino ide, so I don’t have to repeat a manual task by going to a certain directory and deleting the dll file. The dll file gets a wired error access is denied, when the real error is that the file is already running

12:50 AM: I can now generate tabs for the .cpp,.h, keywords.txt, and example file.ino files. I am now getting the string of the file name and file contents to pass to my Arduino class generator.

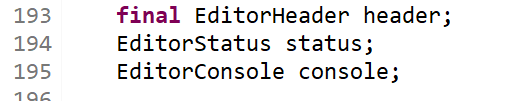
2:41 AM: I can now set the text of the files and save automatically, now I will abstract the code into a method that takes the four strings of the body, text, keywords, and example files, the only unknown inputs

Useful method:

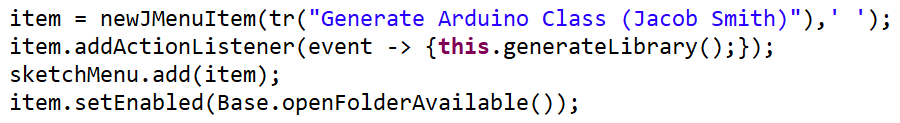
Build method sketch controller.java.



EditorStatus.java line 147, the method used to get the file name, which I want to set automatically



Editor.java line 193 status field which is used to display messages

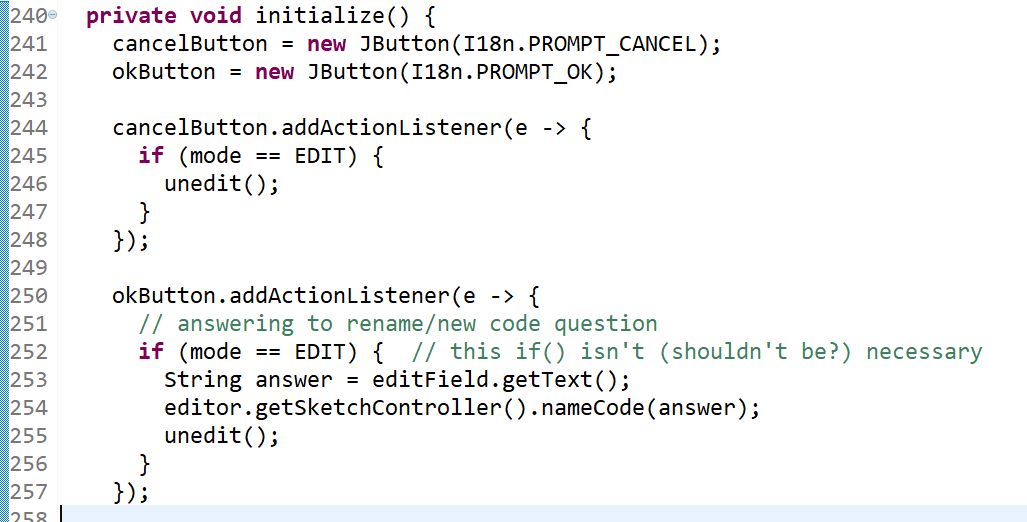


**Arduino IDE showing the new menu option, file name dialogue, and console printout🡪**

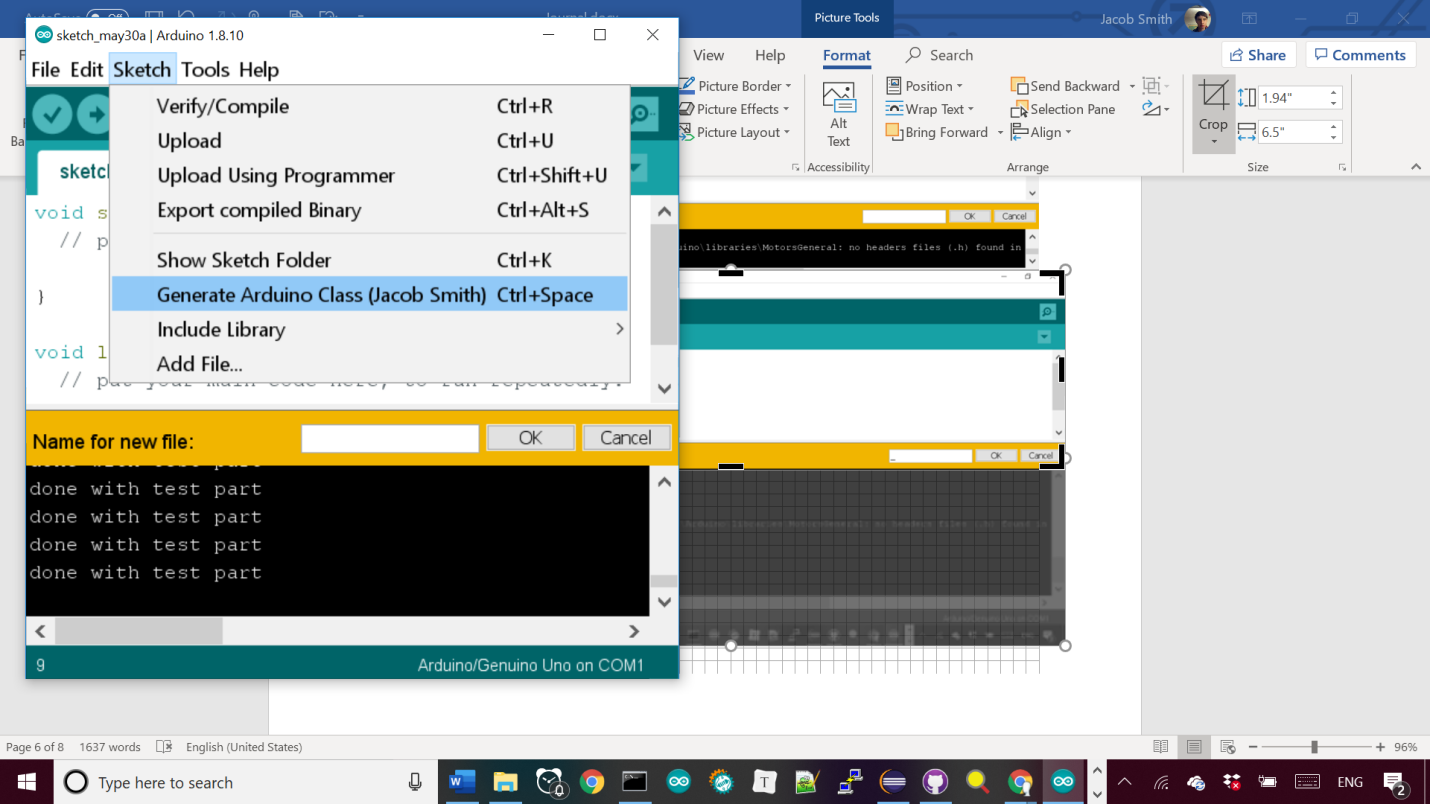
**🡨 Editor Java Line 697, showing how keyboard shortcut finally works**

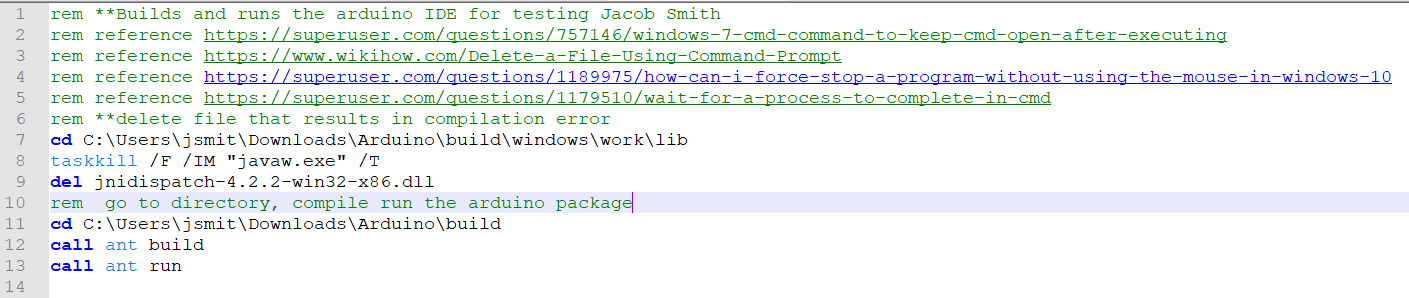
Clockwise from left. These pictures show: how the menu option is shown, how new tabs are created in example IDE code, the menu option being used to display the generate class option which now works with keyboard shortcut, and the batch file that compiles the Arduino application.

This stage of the project was about reverse engineering the Arduino ide to connect it to the class generator API. To do this, I used GREP WIN to search for where the text of a menu option occurred in code to see how it is used and what source functions are important. This allowed me to write my own functions to create tabs for the class information.



**↑Initialize Method of Editor Status.java line 240, shows how ok button is wired to create a new tab**





**RunArduino.bat file to compile and run the Arduino package automatically.**

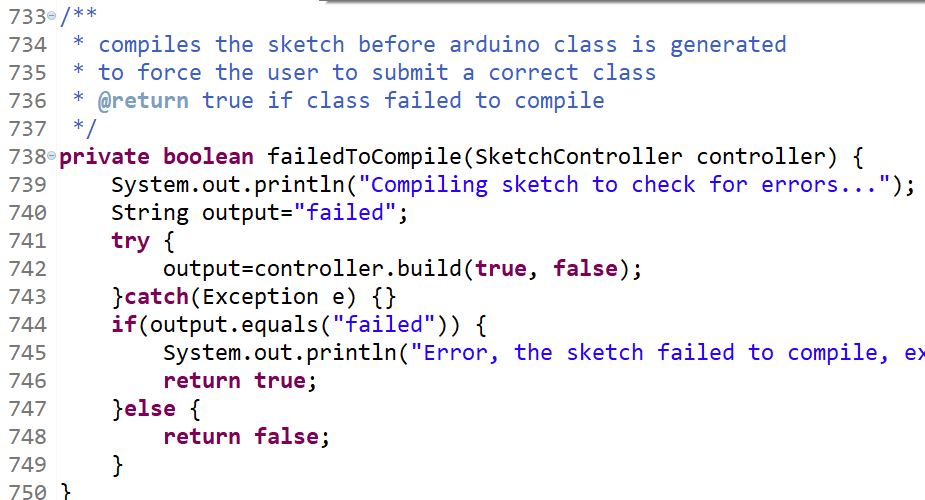
4:46 AM: I wrote this method🡪 to stop library generation if sketch doesn’t compile, it is based on the build method of Sketch Controller class which returns null if the build failed.

Currently, the messages aren’t displayed after the couple seconds it takes to compile the program, probably because I’m not calling a repaint method.

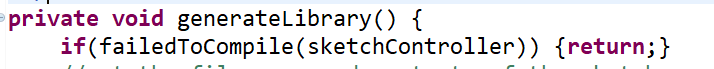
I am working on abstracting the inputs and outputs of the sketch generator interface in the same manner as this method, notice how it is one method that can be used easily and doesn’t rely on global variables.

I’m trying to do this for setting the texts of the tabs, which was complicated because they are automatically alphabetized, but I’m creating them in alphabetical order now.

CODE STYLE: I’m making methods static before I put them in their own class. This will prove that they don’t rely on instance variables, making them easy to put into their own class.



(TOP) My FailedToCompile method in Editor.java to stop generation of library if sketch doesn’t compile. (BOTTOM) usage of method in generate library client code to stop method if compilation fails with screenshots of ide in both use cases.



WORKFLOW: The Arduino IDE lets me input my data directly from the sketch contents and file name and compile the sketch beforehand. This means that I deleted all the classes that prompt the user for input and validates the response. I also separated tests and source files from each other, which will make it easier to put the source files into the Arduino ide, possibly with a jar file to encapsulate and hide them. Potential bug: Some fields can be read to the Arduino class generators as null or "null", I should allow both.

June 3 2019 10:15 PM: After trying out working environments, I settled on this structure**🡪,** where all of the classes to generate an Arduino class from a string are in the ArduinoClassGenerator package, and the interface is in the processing.app class.

The interface needs to be in its own class because it calls the nameCode method of the SketchController class to create tabs, which is a protected variable. **🡪**

The other members of the processing.app class are dummy classes so the referenced classes in the ClassGeneratorInterface don’t result in errors.

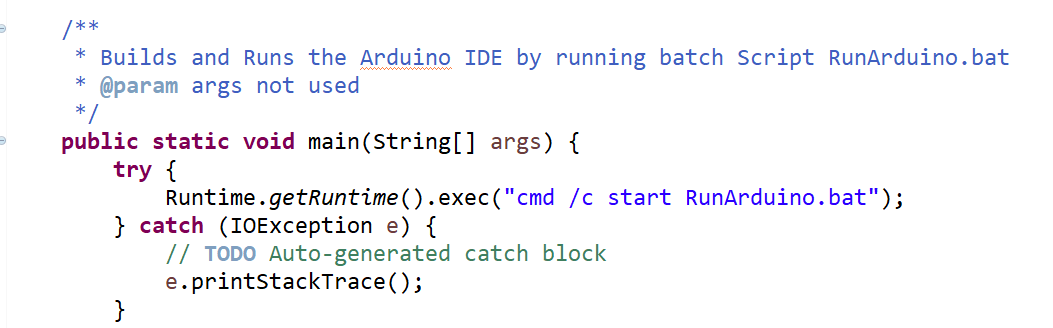
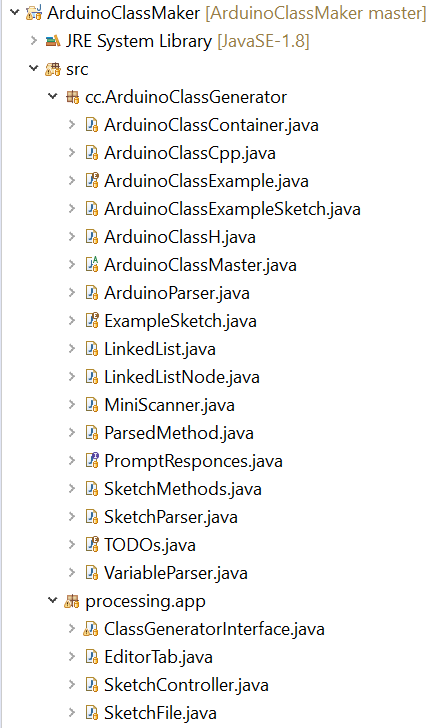
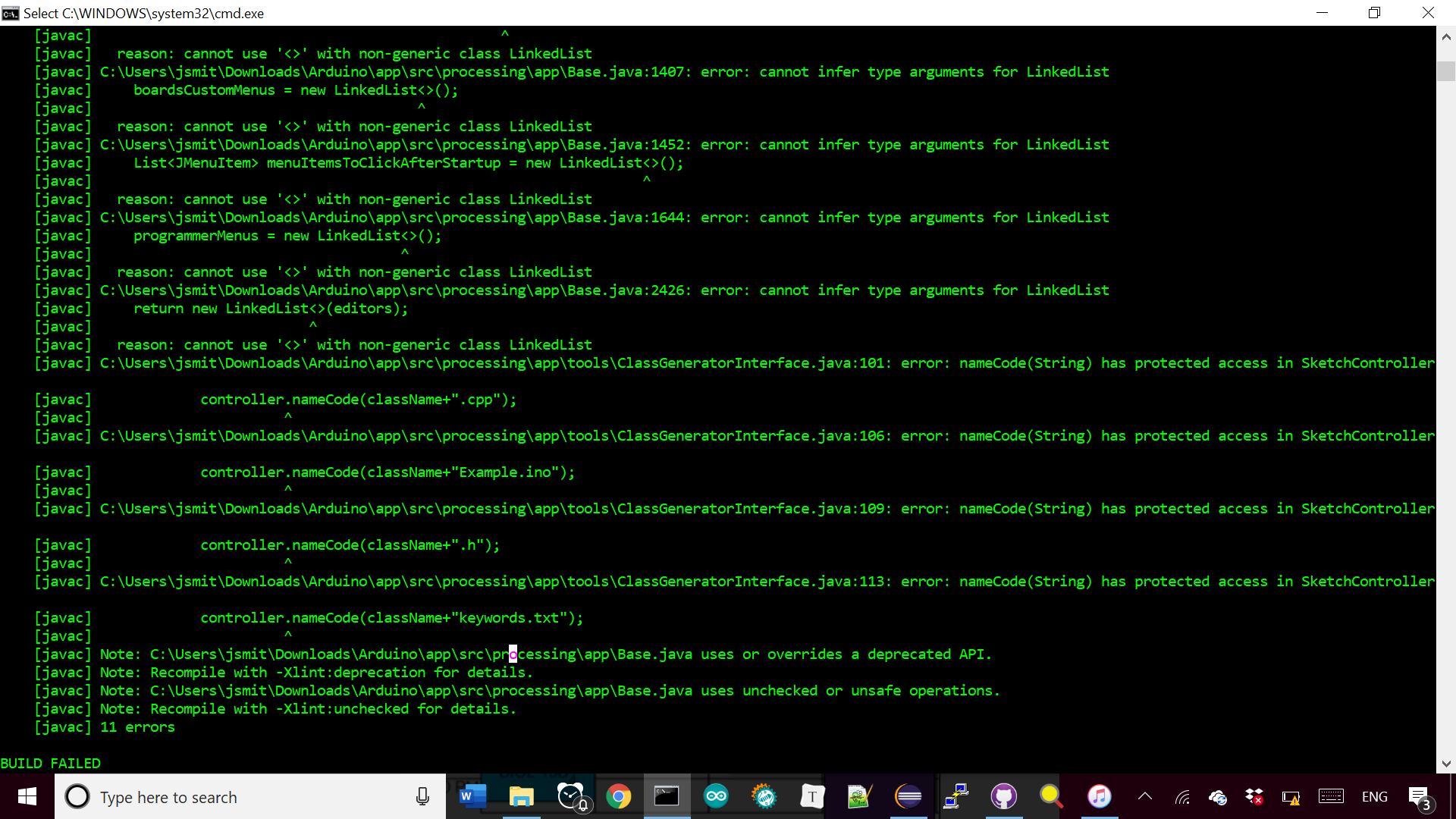
Finally, I can build the arduino IDE with the main method of the ArduinoClassGenerator class, which calls my batch file. **🡪**

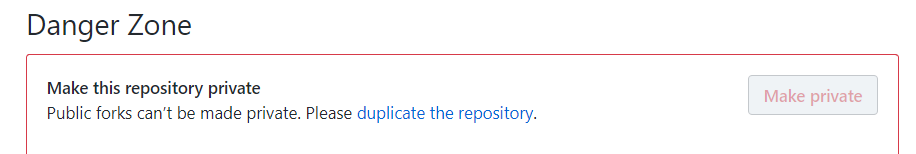
This setup means that all of the required files are in the ArudinoClassGenerator repository (as opposed to having to juggle classes in Arduino build and the batch file being in my desktop to run)

TOP: new Arduino File structure

Below: Error building arduino, sketchController var

Bottom: main method of ArduinoClassGenerator class calling batch file





The batch script is shown at left🡪

Line 15-18 automatically copy files from the ArduinoClassMaker workspace into the Arduino IDE.

Also note the use of function declaration line 33 to reduce redundancy.

The reason I didn’t make my project a fork of the Arduino github project is because I couldn’t import it as an eclipse project and because I can’t make the fork private**🡪**

I looked up using .jar files to package the class generator, but I couldn’t find an example of one being called from java code. Finally, I am including the only other additions to the Arduino IDE necessary to use the class generation feature.

The left shows the Editor class with the code necessary to add class generation as a menu option, which calls the generate library method of the ClassGenerator Interface class.

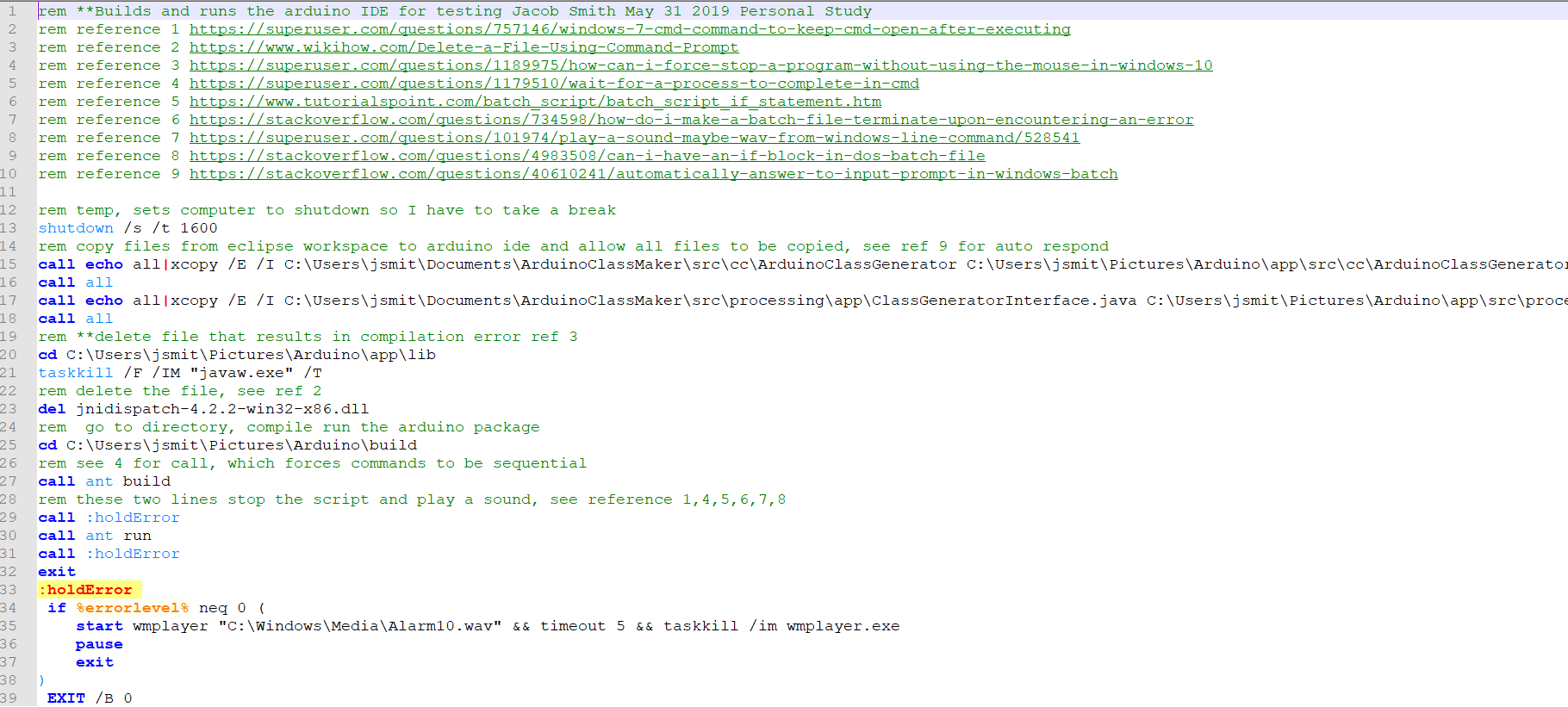
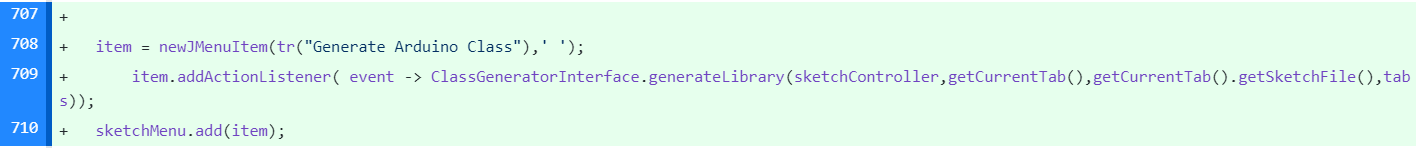
The last modification to the IDE was to allow .txt files🡪, so I can generate the keywords.txt file. This change is unlikely to be accepted by the developers of Arduino, but I could generate the keywords file without opening it.

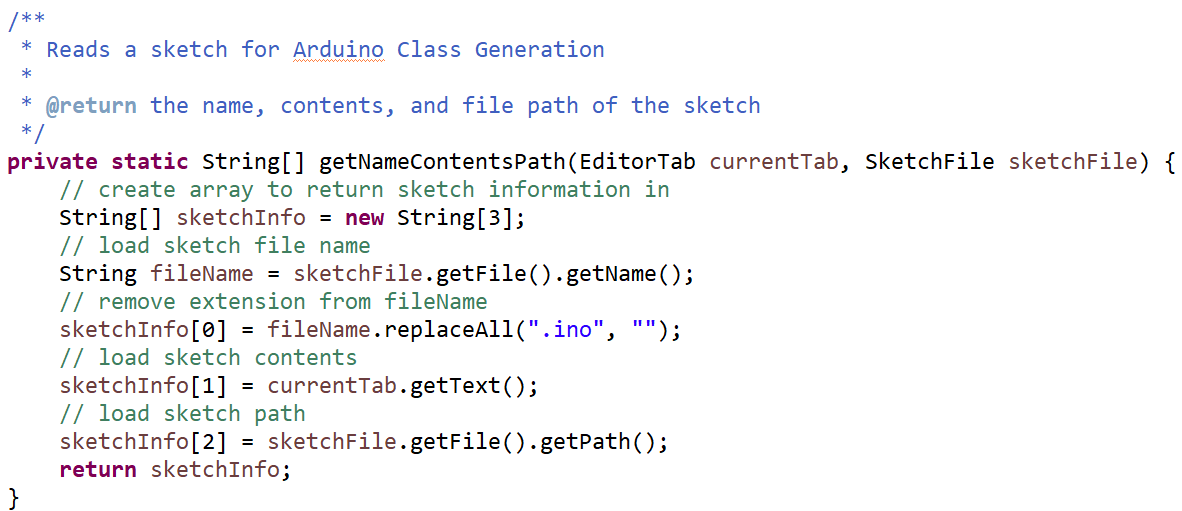
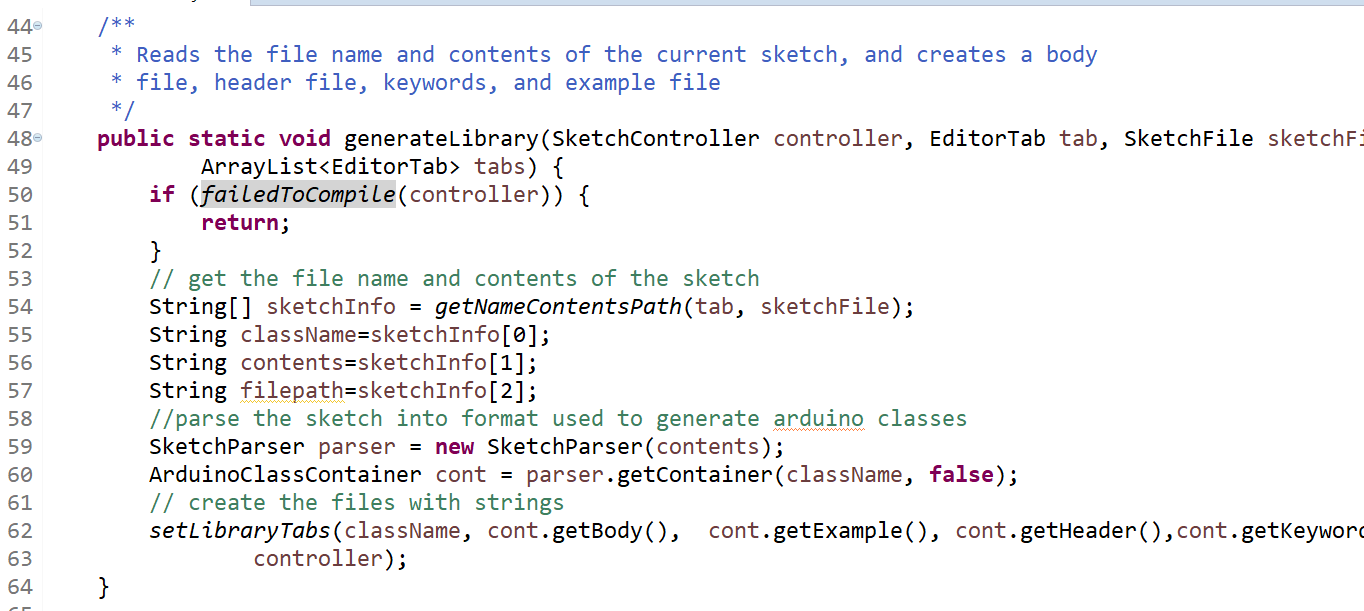


TOP: Github website privacy message

LEFT: Editor.java line 708 showing addition of Line 22 of Sketch.java allowing .txt file for keywords

BOTTOM: Batch file to build Arduino IDE





This is the ClassGenerator Interface class, which is all of the logic necessary to: read the file name and contents of a sketch, generate the class body, header, keywords, and example file, and display them to the IDE.

The failed to Compile method is 3 pages ago.

The getNameContentsPath method gets the filename and contents of the sketch without requiring the user’s input.

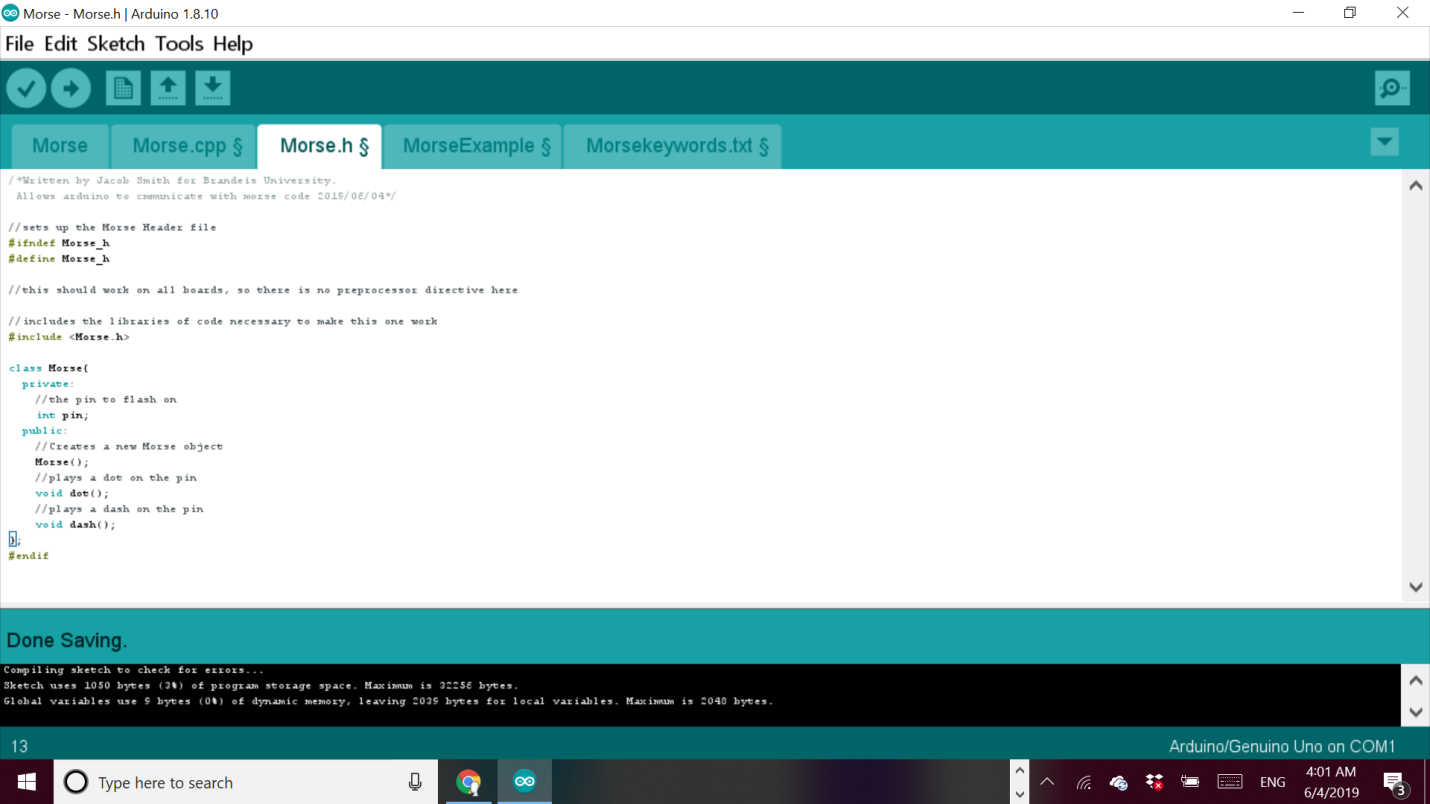
The SetLibraryTabs method actually creates the tabs in the IDE to display the generated class files.

The IDE with current tabs is shown at lower right corner. They are currently not saved automatically so the user can look over them before saving.

FUTURE WORK: A) writing unit tests that cover the generator’s behavior under different types of sketches, B) create interactive user prompts to get the author name, required boards, and organization, display a loading dialogue when class is being generated. C) research whether this project should become a public fork and pull request to Arduino, or whether I should keep it private.

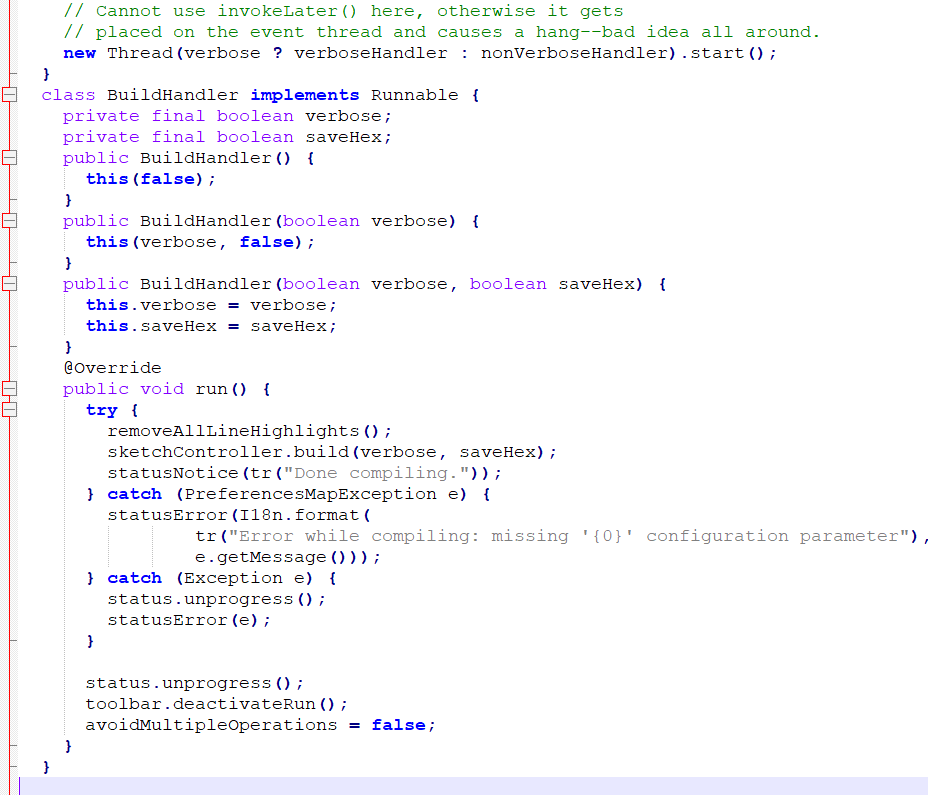
6/10/2019 1:14 PM: Pito Salas documentation, gif creator, gui multithreading, prompt messages.

Because the Class Generator takes a couple seconds to complete, I wanted progress messages to be displayed to show the user something is happening. I can use the progressNotice method of the



TOP: ClassGeneratorInterface Class

LEFT: Arduino IDE with generated class information tabs



Editor status method, but that only displays the messages at the end of execution. The reason is because graphical user interfaces are multithreaded. To find a solution, I found this source from codeRanch:

1. <https://coderanch.com/wiki/660058/JProgressBar-Doesnt-Update>

And this on Worker Threads:

1. <https://docs.oracle.com/javase/tutorial/uiswing/concurrency/worker.html>
2. [https://docs.oracle.com/javase/tutorial/displayCode.html?code=https://docs.oracle.com/javase/tutorial/uiswing/examples/concurrency/FlipperProject/src/concurrency/Flipper.java](https://docs.oracle.com/javase/tutorial/displayCode.html?code=https:/docs.oracle.com/javase/tutorial/uiswing/examples/concurrency/FlipperProject/src/concurrency/Flipper.java)

I simplified the code from sources 1 and 3, but in the end I looked up how displaying is handles by the Arduino IDE (TOP)--> , and I saw how the Editor class is able to display interm results while compiling.

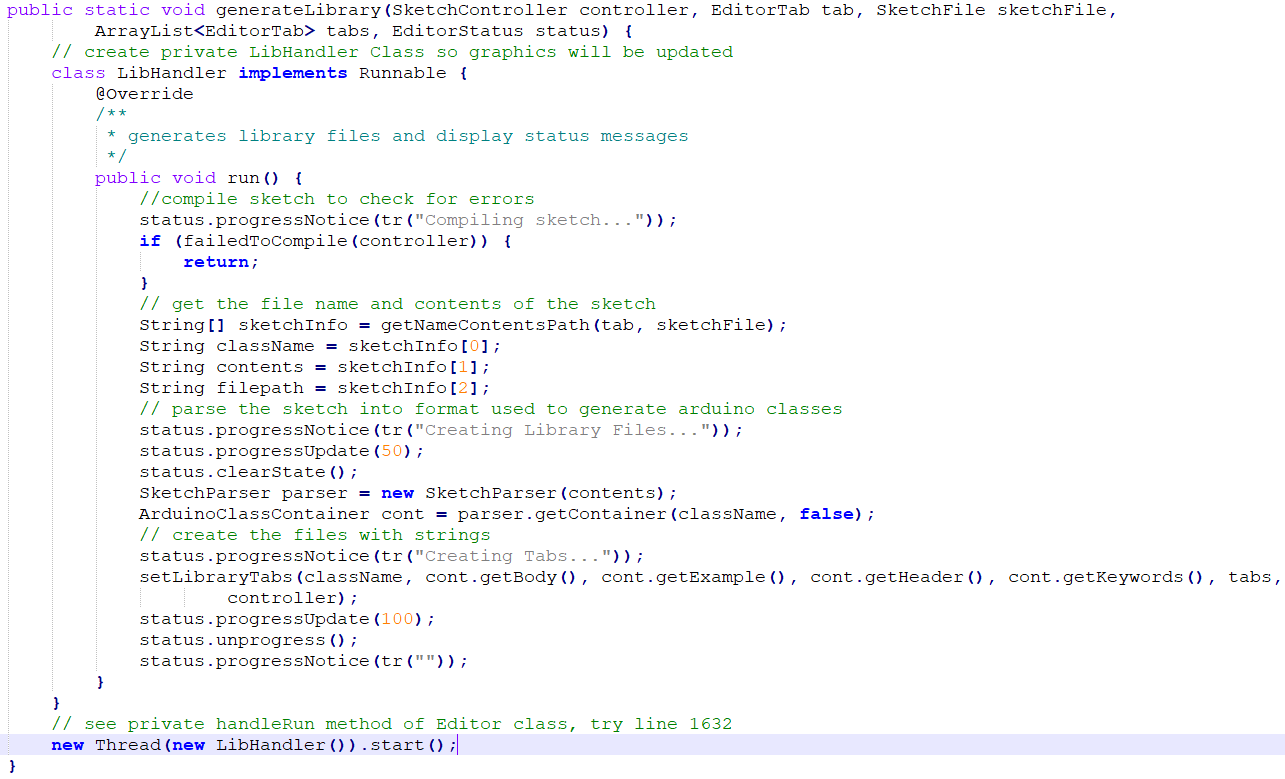
This method is to create an internal class that implements the Runnable interface, with a run method which performs computations and displays prompts, and then create a new Thread which calls the start method.

This approach makes sense, because it creates a thread other than the display thread to perform computations on, which was the cause of my problem.

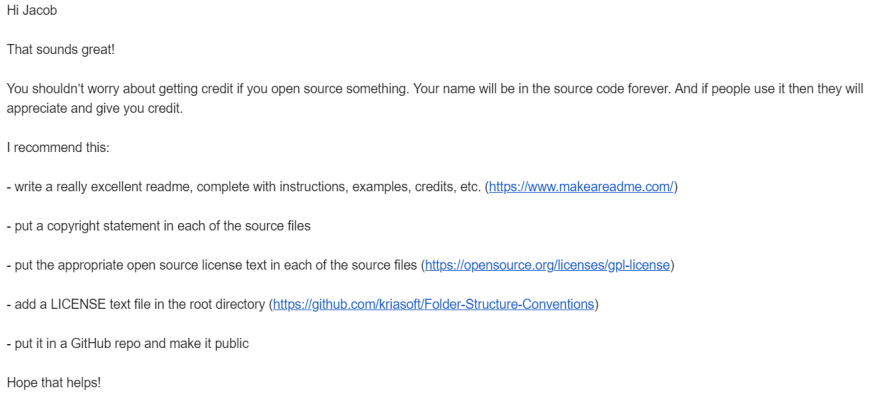
I created the internal libHandler class, where the run method displays prompts just like at top of page, and the class generator now displays the prompts properly.

(TOP) Editor Line 1630, Showing how compilation is able to display messages while thread is being compiled.

(Bottom) GenerateLibrary method of ClassGeneratorInterface, showing additions to display interm prompts to the user.



GenerateLibrary method of ArduinoClassGenerator. The LibHandler Class to show how to display interm messages in Arduino IDE, the new Thread line executes the run method.



(TOP) Documentation from Professor Salas

DOCUMENTATION: I got an email from Professor Salas on how to better document the github page,--> and one useful feature was that I can add gifs in my readMe

Screen Capture to make readme[1]:<https://www.laptopmag.com/articles/how-to-video-screen-capture-windows-10>

Convert Video to GIF: [2]<https://giphy.com/>

Add Gifs to github readme: [3]<https://medium.com/@josephcardillo/how-to-add-gifs-to-your-github-readme-89c74da2ce47>

<https://softwareengineering.stackexchange.com/questions/19649/copyright-notices-disclaimers-in-source-files>