Developing a Tanaguru Testing Framework

By: Patrick Amons, Hollande Powell, Montrel Nesbitt and Grant Jackson

Table of Contents

Introduction

Our overall goal this semester was to develop a testing framework for an open source project that scales easily as the project grows. Initially we had a list of roughly 30 open source projects to choose from and as you'll see in our next section we were able to narrow our search down to 3 prime candidates. Throughout the rest of this document, we will cover why we chose Tanaguru to be the project we pursued, the aspects of testing, the framework of our testing script, the revisions that we've made to our initial framework, the injecting of faults into our methods to insure proper testing, and our experience in developing this project over the semester.

Chapter 1

Beach Boys

Patrick Amons
Grant Jackson
Hollande Powell
Montrel Nesbitt

Candidate 1: Drone 4 Dengue

Overview

Drone 4 Dengue is a software project that uses drone images to detect dengue mosquito breeding sites. These particular mosquitoes can transmit a disease called Dengue Fever, this disease is primarily transmitted in subtropical areas of the world.

Running and compiling

Unfortunately, this project has not been updated in many years and we kept running into compatibility issues when trying to compile and run it. Ultimately due to these issues, we decided not to go with this project.

Conclusion

Ultimately due to the issues listed above, we decided not to go with this project.

Candidate 2: Epidemiological Modeler (STEM)

Overview

The Epidemiological Modeler (STEM) is a software tool developed to help scientists and public health officials create models and run simulations of emerging infectious diseases. These simulations can help understand and prevent diseases before they could possibly become a worldwide pandemic. We found this project interesting because we thought it related to what's going on in the world today.

Running and compiling

This project requires the Eclipse IDE in order to run along with several other Eclipse plugins. Although this project does have a useful setup page, it is lacking some information and made compiling and running this project a challenge. The first challenge we encountered was that this project requires quite a lot of plugins in order to run. The issue was standard Eclipse does not ship with the plug-in development environment, so when we tried to install the STEM plugins the "Plug-in Development" option under preferences was not there. After some research, we found out we had to install the plug-in development environment from the Eclipse Marketplace. After quite a bit of time, the project finished getting the required plug-ins and the number of errors went from a few thousand down to four. In order to resolve these four errors, we used Eclipse's quick-fix option to install one additional library. After installing this library, recompiling the code, and completing the instructions on the setup page the program ran without any issues.

Conclusion

Although we did get this project to run without any errors, in order to run a simulation we would have to know quite a lot about pathology and epidemiology. Something none of us on this team has any knowledge of. Due to this steep learning curve we decided to not go with this project.

Candidate 3: Tanaguru Contrast Finder

Overview

Tanaguru Contrast Finder assesses the background color contrast of a website in order to make websites more accessible to people with color blindness. Tanaguru also develops an open-source website assessment tool that uses apps to run tests on a website, the Contrast Finder is one of these apps that Tanaguru develops themselves.

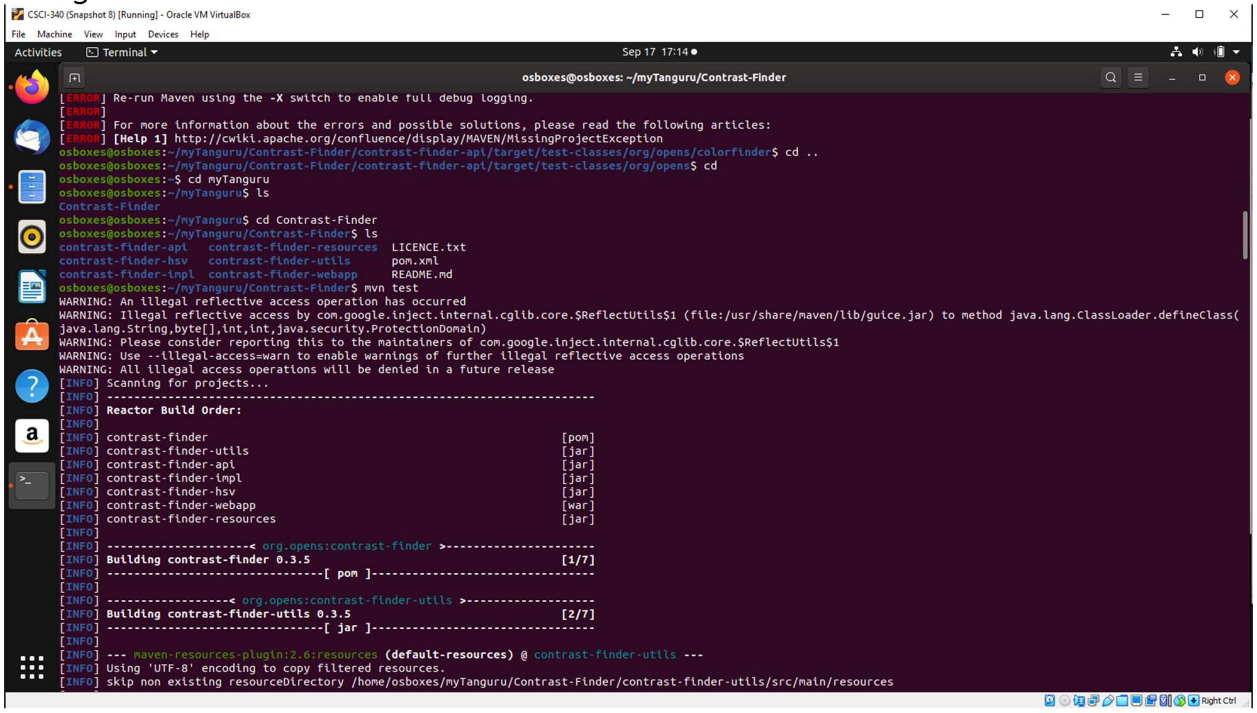
Running and compiling

This project required some experimentation in order to get it to run. Since it is an app that runs on the website assessment tool we had to figure out whether or not we should try to get the app to run by itself or if we should attempt to get the assessment tool running and test it from there. After some trial and error, we found out that the project builder Maven is required to compile the contrast finder. Once we found this out, we started to look into how to use Maven and found out in order to build this project we had to use the command "mvn test" from the command line in the Contrast Finder directory. After some time we got the Contrast Finder to build.

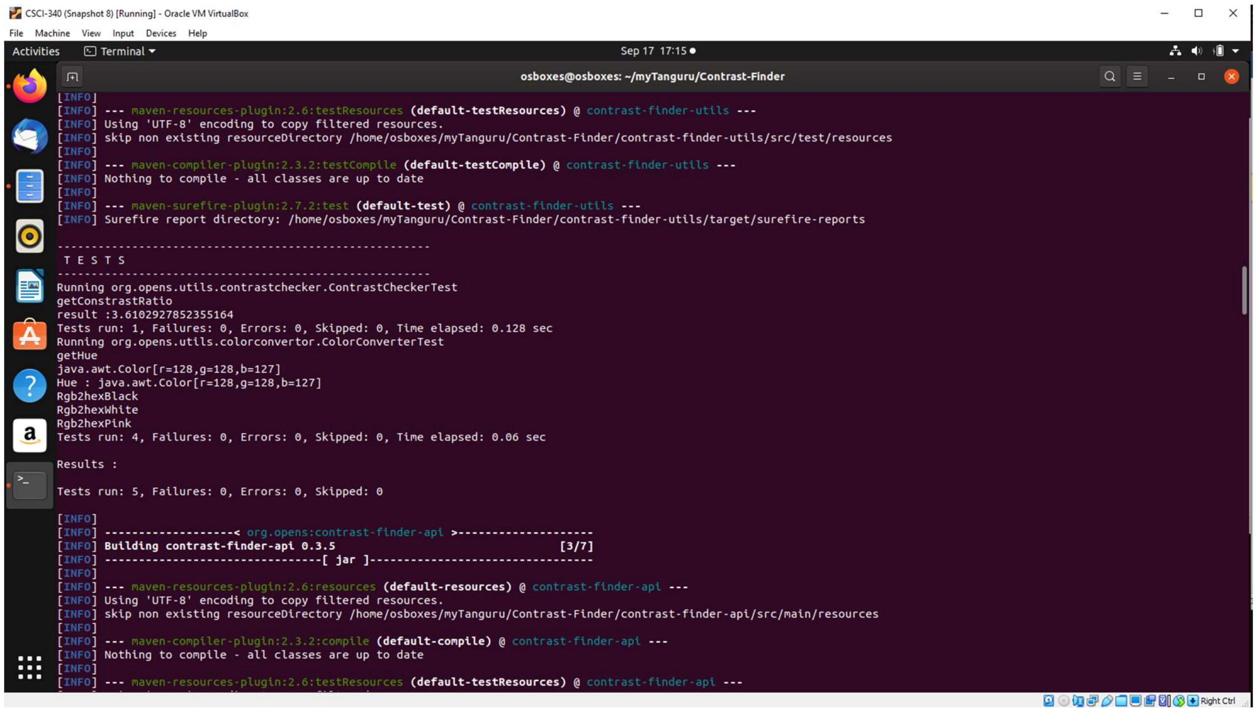
Conclusion

Ultimately we decided to choose this project because we got it to successfully build and it seems fairly simple to use, compared to STEM. Some screenshots are included below of building and running Contrast Finder

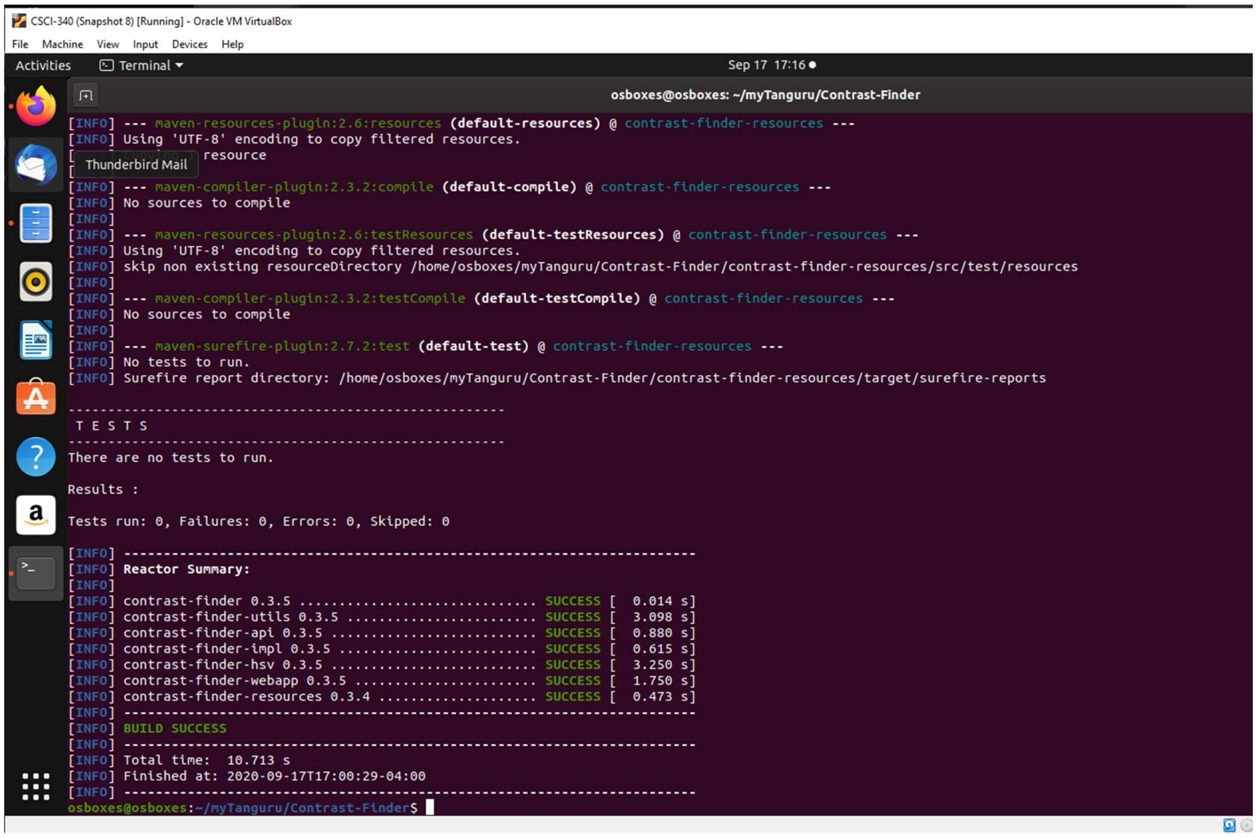
Screenshots building and running tests for
Tanaguru


CSCI-340 (Snapshot 8) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Sep 17 17:14
osboxes@osboxes: ~/myTanguru/Contrast-Finder
Right Ctrl

```
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/MissingProjectException
osboxes@osboxes:~/myTanguru/Contrast-Finder/contrast-finder-api/target/test-classes/org/opens/colorfinder$ cd ..
osboxes@osboxes:~/myTanguru/Contrast-Finder/contrast-finder-api/target/test-classes/org/opens$ cd
osboxes@osboxes:~$ cd myTanguru
osboxes@osboxes:~/myTanguru$ ls
Contrast-Finder
osboxes@osboxes:~/myTanguru$ cd Contrast-Finder
osboxes@osboxes:~/myTanguru/Contrast-Finder$ ls
contrast-finder-api   contrast-finder-resources  LICENCE.txt
contrast-finder-hsv   contrast-finder-utils      pom.xml
contrast-finder-impl  contrast-finder-webapp     README.md
osboxes@osboxes:~/myTanguru/Contrast-Finder$ mvn test
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by com.google.inject.internal.cglib.core.$ReflectUtils$1 (file:/usr/share/maven/lib/guice.jar) to method java.lang.ClassLoader.defineClass(
java.lang.String,byte[],int,int,java.security.ProtectionDomain)
WARNING: Please consider reporting this to the maintainers of com.google.inject.internal.cglib.core.$ReflectUtils$1
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
[INFO] Scanning for projects...
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Build Order:
[INFO]
[INFO] contrast-finder                                                    [pom]
[INFO] contrast-finder-utils                                              [jar]
[INFO] contrast-finder-api                                                [jar]
[INFO] contrast-finder-impl                                               [jar]
[INFO] contrast-finder-hsv                                                [jar]
[INFO] contrast-finder-webapp                                             [war]
[INFO] contrast-finder-resources                                          [jar]
[INFO]
[INFO] --------------------< org.opens:contrast-finder >---------------------
[INFO] Building contrast-finder 0.3.5                                     [1/7]
[INFO] --------------------------------[ pom ]---------------------------------
[INFO]
[INFO] -----------------< org.opens:contrast-finder-utils >------------------
[INFO] Building contrast-finder-utils 0.3.5                               [2/7]
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ contrast-finder-utils ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/osboxes/myTanguru/Contrast-Finder/contrast-finder-utils/src/main/resources
```


CSCI-340 (Snapshot 8) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Sep 17 17:15
osboxes@osboxes: ~/myTanguru/Contrast-Finder
Right Ctrl

```
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ contrast-finder-utils ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/osboxes/myTanguru/Contrast-Finder/contrast-finder-utils/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:2.3.2:testCompile (default-testCompile) @ contrast-finder-utils ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.7.2:test (default-test) @ contrast-finder-utils ---
[INFO] Surefire report directory: /home/osboxes/myTanguru/Contrast-Finder/contrast-finder-utils/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.opens.utils.contrastchecker.ContrastCheckerTest
getConstrastRatio
result :3.6102927852355164
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.128 sec
Running org.opens.utils.colorconvertor.ColorConverterTest
getHue
java.awt.Color[r=128,g=128,b=127]
Hue : java.awt.Color[r=128,g=128,b=127]
Rgb2hexBlack
Rgb2hexWhite
Rgb2hexPink
Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.06 sec

Results :

Tests run: 5, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] ------------------< org.opens:contrast-finder-api >-------------------
[INFO] Building contrast-finder-api 0.3.5                                [3/7]
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ contrast-finder-api ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/osboxes/myTanguru/Contrast-Finder/contrast-finder-api/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:2.3.2:compile (default-compile) @ contrast-finder-api ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ contrast-finder-api ---
```


CSCI-340 (Snapshot 8) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Sep 17 17:16
osboxes@osboxes: ~/myTanguru/Contrast-Finder
Thunderbird Mail

```
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ contrast-finder-resources ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[                 resource
[
[INFO] --- maven-compiler-plugin:2.3.2:compile (default-compile) @ contrast-finder-resources ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ contrast-finder-resources ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/osboxes/myTanguru/Contrast-Finder/contrast-finder-resources/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:2.3.2:testCompile (default-testCompile) @ contrast-finder-resources ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-surefire-plugin:2.7.2:test (default-test) @ contrast-finder-resources ---
[INFO] No tests to run.
[INFO] Surefire report directory: /home/osboxes/myTanguru/Contrast-Finder/contrast-finder-resources/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------
There are no tests to run.

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] contrast-finder 0.3.5 .............................. SUCCESS [  0.014 s]
[INFO] contrast-finder-utils 0.3.5 ........................ SUCCESS [  3.098 s]
[INFO] contrast-finder-api 0.3.5 .......................... SUCCESS [  0.880 s]
[INFO] contrast-finder-impl 0.3.5 ......................... SUCCESS [  0.615 s]
[INFO] contrast-finder-hsv 0.3.5 .......................... SUCCESS [  3.250 s]
[INFO] contrast-finder-webapp 0.3.5 ....................... SUCCESS [  1.750 s]
[INFO] contrast-finder-resources 0.3.4 .................... SUCCESS [  0.473 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  10.713 s
[INFO] Finished at: 2020-09-17T17:00:29-04:00
[INFO] ------------------------------------------------------------------------
osboxes@osboxes:~/myTanguru/Contrast-Finder$
```

Chapter 2

Test Plan

Testing Process:

We will be testing multiple methods in the ColorConverter.java class within Tanaguru's system. To do this, we will be constructing driver classes for each method being tested, then feeding information into the drivers through a script that will read through our test case files and compare the driver output to the expected output.

Requirement Traceability:

getBrightness requires a Color class as input and returns a float value. Test Cases 1-5

getSaturation requires a Color class as input and returns a float value. Test Cases 6-10

getHue requires a Color class as input and returns a float value. Test Cases 11-15

rgb2Hex requires a Color class as input and returns a string. Test Cases 16-20

hex2Rgb requires a hex string of a color as input and returns a string. Test Cases 21-25

Tested Items:

The tests will be on getBrightness(Color color), getSaturation(Color color), getHue(Color color), rgb2Hex(Color color), and hex2Rgb(String colorStr).

Testing Schedule:

Every Tuesday and Thursday 4:00pm - 4:50pm

Test Recording Procedures:

Tests results will be fed into an html file and then displayed in a browser for easy readability and storage.

Hardware and software Requirements:

- Linux System or VirtualBox 6.1 - https://www.virtualbox.org/
- Git - https://git-scm.com/downloads
- Maven - https://maven.apache.org/download.cgi
- OpenJDK 14.01 - https://jdk.java.net/14/
- Tanaguru - https://github.com/Tanaguru/Contrast-Finder

Constraints:

Limited amount of time. Unfamiliarity with the Java color class and Tanaguru system. Limited team size.

Test Cases:

Click here to view test case files.

Test Case Template:
Line 1: Test Case ID #
Line 2: Description of method
Line 3: Class Name
Line 4: Driver Name
Line 5: Driver Input
Line 6: Driver Expected Output
Line 7: Method Name

Test Case Expected Outputs:
getBrightness will return 50.0 when the color (GREEN) is input
getBrightness will return 100.0 when the color (RED) is input
getBrightness will return 50.0 when the color (BLUE) is input
getBrightness will return 0.0 when the color (BLACK) is input
getBrightness will return 50.2 when the color (PURPLE) is input

getSaturation will return 0.0 when the color (BLACK) is input
getSaturation will return 0.0 when the color (WHITE) is input
getSaturation will return 1.0 when the color (RED) is input
getSaturation will return 1.0 when the color (YELLOW) is input
getSaturation will return 0.946 when a custom color (R: 13 G: 242 B: 135) is input

getHue will return 0.0 when the color (RED) is input
getHue will return 120.0 when the color (GREEN) is input
getHue will return 240.0 when the color (BLUE) is input
getHue will return 0.0 when the color (BLACK) is input
getHue will return 300.0 when the color (PURPLE) is input

rgb2Hex will return #0000FF when the color (0, 0, 255)(BLUE) is input
rgb2Hex will return #00FF00 when the color (0, 255, 0)(GREEN) is input
rgb2Hex will return #FF0000 when the color (255, 0, 0)(RED) is input
rgb2Hex will return #000000 when the color (0, 0, 0)(BLACK) is input
rgb2Hex will return #800080 when the color (128, 0, 128)(PURPLE) is input

hex2Rgb will return (255, 0, 0) when the color #FF0000 (RED) is input
hex2Rgb will return (0, 255, 0) when the color #00FF00 (GREEN) is input
hex2Rgb will return (0, 0, 255) when the color #0000FF (BLUE) is input
hex2Rgb will return (0, 0, 0) when the color #000000(BLACK) is input
hex2Rgb will return (128, 0, 128) when the color #800080(PURPLE) is input

Chapter 3

Testing Framework

ARCHITECTURAL DESCRIPTION

- Our framework starts with compiling all of our test case drivers (one for each method being tested) and then it creates an empty html file to store our results in. As our script runs through all of the files in the testCases directory it reads in each line of the file and then locally stores the line as a variable. The file being read from then gets it information passed into the drivers which outputs its processed information into a table within the html file. Results are then compared to their expected outcomes in the html file to determine if the test passed or not.

HOW TO DOCUMENTATION

Step 1: Install jdk14

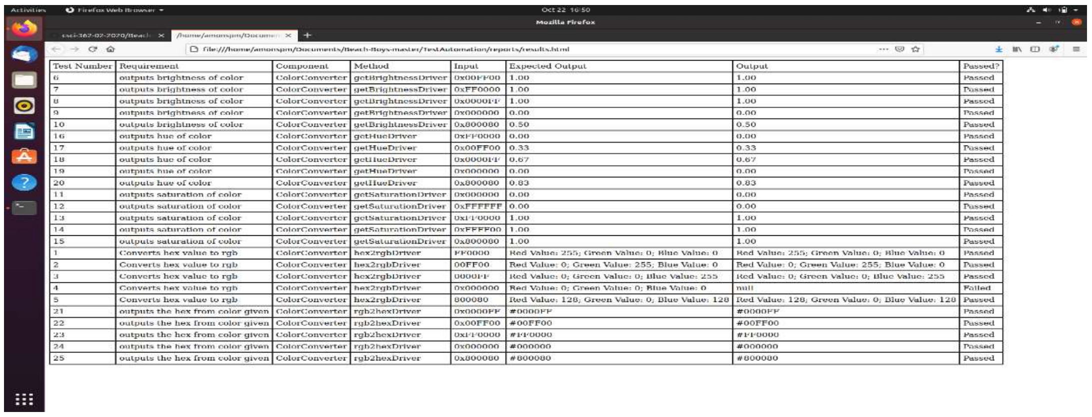
Step 2: Clone the repository @ https://github.com/csci-362-02-2020/Beach-Boys

Step 3: Navigate to the test automation directory in the terminal

Step 4: Run the make file by typing 'make' in the command line

TEST CASES

All 25 test cases can be found in the testCases folder within the code section of our team github



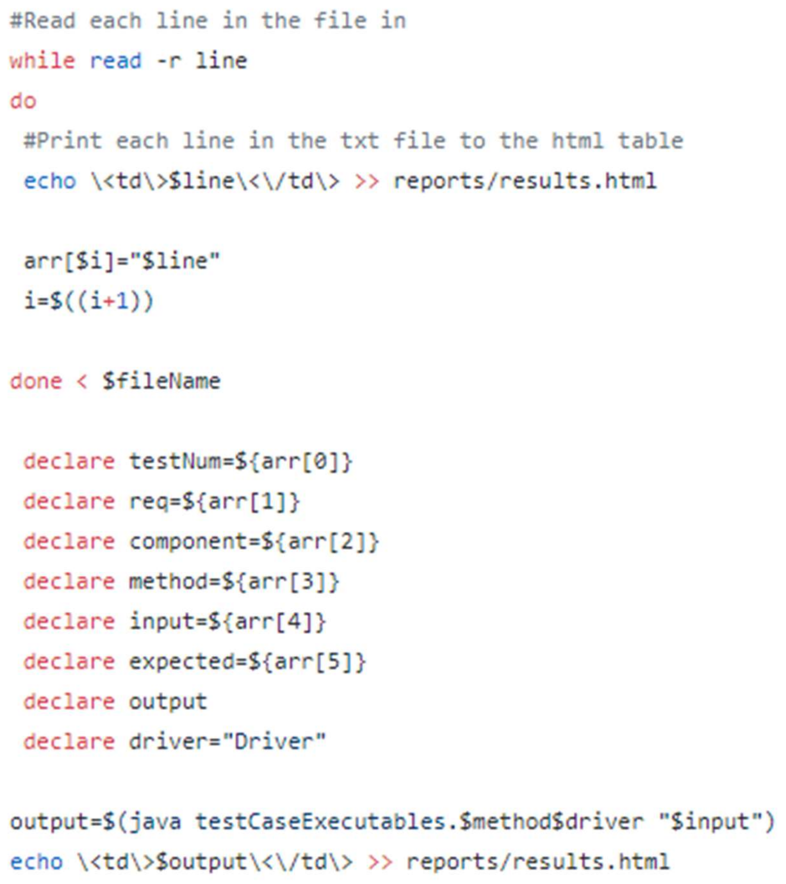
Test Number	Requirement	Component	Method	Input	Expected Output	Output	Passed?
6	outputs brightness of color	ColorConverter	getBrightnessDriver	0x00FF00	1.00	1.00	Passed
7	outputs brightness of color	ColorConverter	getBrightnessDriver	0xFF0000	1.00	1.00	Passed
8	outputs brightness of color	ColorConverter	getBrightnessDriver	0x0000FF	1.00	1.00	Passed
9	outputs brightness of color	ColorConverter	getBrightnessDriver	0x000000	0.00	0.00	Passed
10	outputs brightness of color	ColorConverter	getBrightnessDriver	0x800080	0.50	0.50	Passed
16	outputs hue of color	ColorConverter	getHueDriver	0xFF0000	0.00	0.00	Passed
17	outputs hue of color	ColorConverter	getHueDriver	0x00FF00	0.33	0.33	Passed
18	outputs hue of color	ColorConverter	getHueDriver	0x0000FF	0.67	0.67	Passed
19	outputs hue of color	ColorConverter	getHueDriver	0x000000	0.00	0.00	Passed
20	outputs hue of color	ColorConverter	getHueDriver	0x800080	0.83	0.83	Passed
11	outputs saturation of color	ColorConverter	getSaturationDriver	0x000000	0.00	0.00	Passed
12	outputs saturation of color	ColorConverter	getSaturationDriver	0xFFFFFF	0.00	0.00	Passed
13	outputs saturation of color	ColorConverter	getSaturationDriver	0xFF0000	1.00	1.00	Passed
14	outputs saturation of color	ColorConverter	getSaturationDriver	0xFFFF00	1.00	1.00	Passed
15	outputs saturation of color	ColorConverter	getSaturationDriver	0x800080	1.00	1.00	Passed
1	Converts hex value to rgb	ColorConverter	hex2rgbDriver	FF0000	Red Value: 255; Green Value: 0; Blue Value: 0	Red Value: 255; Green Value: 0; Blue Value: 0	Passed
2	Converts hex value to rgb	ColorConverter	hex2rgbDriver	00FF00	Red Value: 0; Green Value: 255; Blue Value: 0	Red Value: 0; Green Value: 255; Blue Value: 0	Passed
3	Converts hex value to rgb	ColorConverter	hex2rgbDriver	0000FF	Red Value: 0; Green Value: 0; Blue Value: 255	Red Value: 0; Green Value: 0; Blue Value: 255	Passed
4	Converts hex value to rgb	ColorConverter	hex2rgbDriver	0x000000	Red Value: 0; Green Value: 0; Blue Value: 0	null	Failed
5	Converts hex value to rgb	ColorConverter	hex2rgbDriver	800080	Red Value: 128; Green Value: 0; Blue Value: 128	Red Value: 128; Green Value: 0; Blue Value: 128	Passed
21	outputs the hex from color given	ColorConverter	rgb2hexDriver	0x0000FF	#0000FF	#0000FF	Passed
22	outputs the hex from color given	ColorConverter	rgb2hexDriver	0x00FF00	#00FF00	#00FF00	Passed
23	outputs the hex from color given	ColorConverter	rgb2hexDriver	0xFF0000	#FF0000	#FF0000	Passed
24	outputs the hex from color given	ColorConverter	rgb2hexDriver	0x000000	#000000	#000000	Passed
25	outputs the hex from color given	ColorConverter	rgb2hexDriver	0x800080	#800080	#800080	Passed

Chapter 4

Revisions of Framework

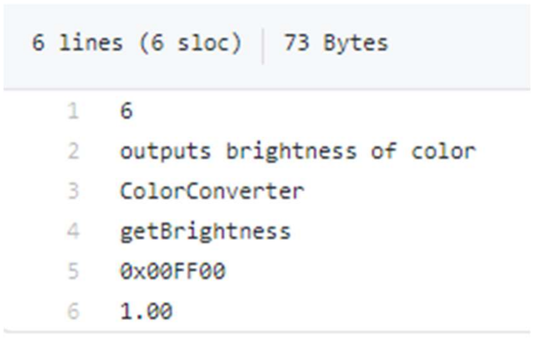
Changes:

- We have recently modified the main script to know nothing about the drivers
- We also modified the test case files to pass in the method name directly. The script now takes in the method name and finds a driver suitable for it to run in.



```
#Read each line in the file in
while read -r line
do
 #Print each line in the txt file to the html table
 echo \<td\>$line\<\/td\> >> reports/results.html

 arr[$i]="$line"
 i=$((i+1))

done < $fileName

 declare testNum=${arr[0]}
 declare req=${arr[1]}
 declare component=${arr[2]}
 declare method=${arr[3]}
 declare input=${arr[4]}
 declare expected=${arr[5]}
 declare output
 declare driver="Driver"

output=$(java testCaseExecutables.$method$driver "$input")
echo \<td\>$output\<\/td\> >> reports/results.html
```

The above picture shows our bash script reading in the test cases and at the bottom is the new implementation of finding the correct driver to run. Previously there was an if statement here comparing the method with each of the drivers to see if it matched.



```
6 lines (6 sloc) | 73 Bytes
1  6
2  outputs brightness of color
3  ColorConverter
4  getBrightness
5  0x00FF00
6  1.00
```

Here is a picture of the current test cases. We have changed line 4 of each test case to now have the method name being tested instead of the driver name for the method.

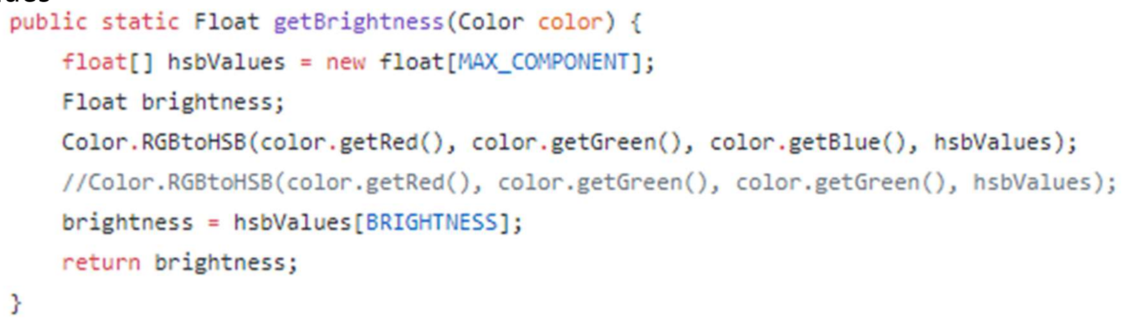
Chapter 5

Injecting Faults

All changes take place in the ColorConverter.java method

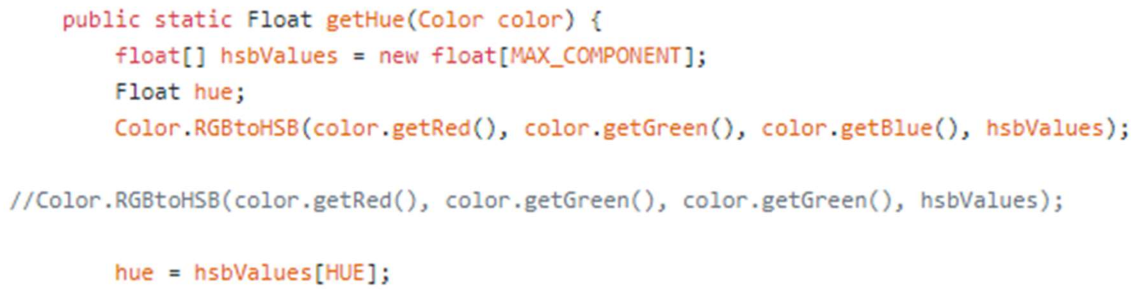
For each of the methods we have our faults commented out and they will need to be uncommented and then the corresponding correct line needs to be commented out.

Line 79 - Change getBrightness by passing it the wrong RGB values



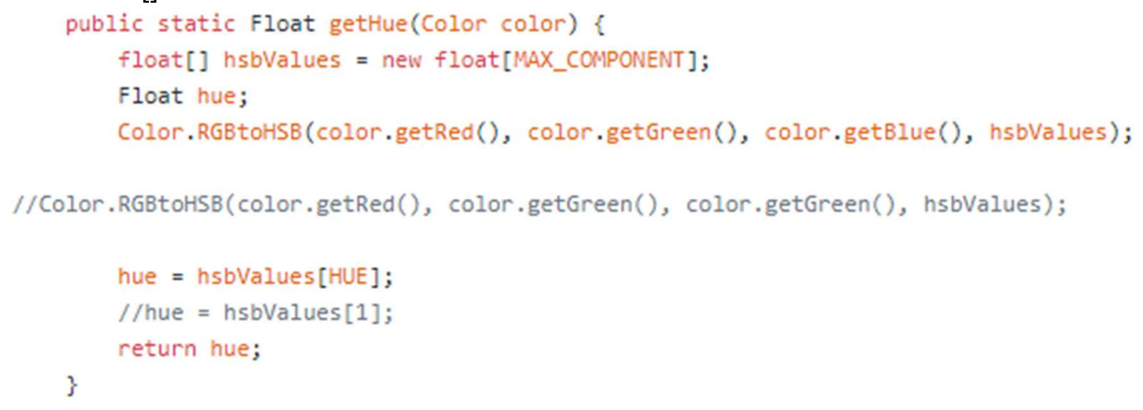
```
public static Float getBrightness(Color color) {
    float[] hsbValues = new float[MAX_COMPONENT];
    Float brightness;
    Color.RGBtoHSB(color.getRed(), color.getGreen(), color.getBlue(), hsbValues);
    //Color.RGBtoHSB(color.getRed(), color.getGreen(), color.getGreen(), hsbValues);
    brightness = hsbValues[BRIGHTNESS];
    return brightness;
}
```

Line 107 - Change getHue by passing it the wrong RGB values



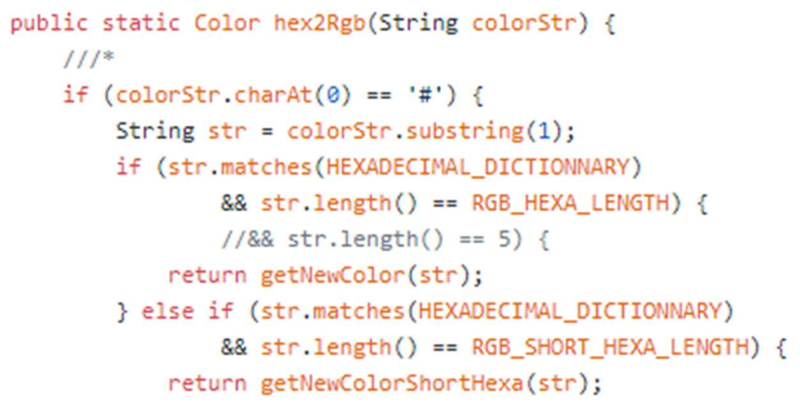
```
    public static Float getHue(Color color) {
        float[] hsbValues = new float[MAX_COMPONENT];
        Float hue;
        Color.RGBtoHSB(color.getRed(), color.getGreen(), color.getBlue(), hsbValues);

//Color.RGBtoHSB(color.getRed(), color.getGreen(), color.getGreen(), hsbValues);

        hue = hsbValues[HUE];
```

Line 110 - Change getHue by searching in the wrong index of hsbValues[]



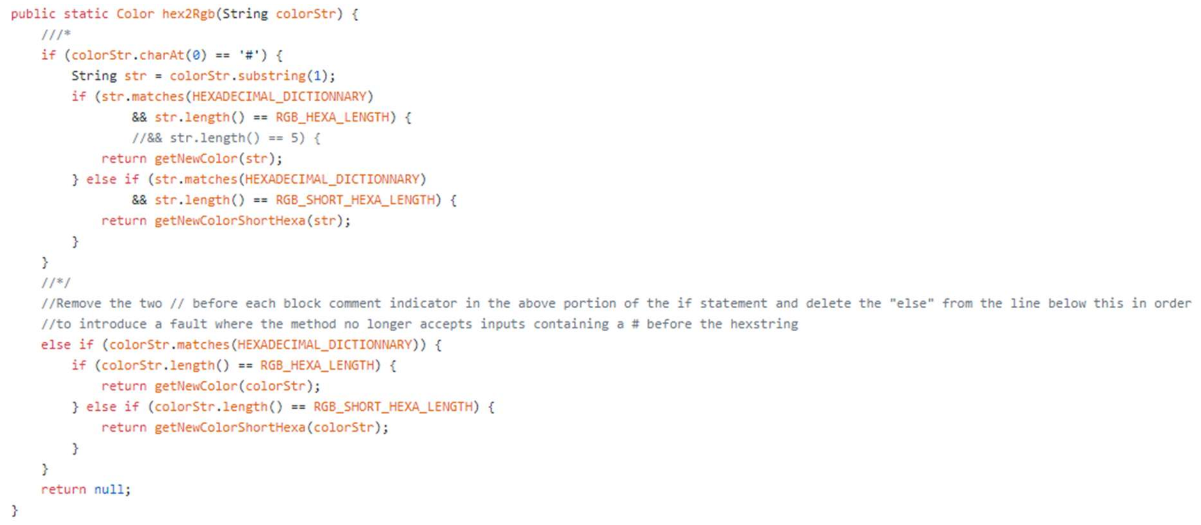
```
public static Float getHue(Color color) {
    float[] hsbValues = new float[MAX_COMPONENT];
    Float hue;
    Color.RGBtoHSB(color.getRed(), color.getGreen(), color.getBlue(), hsbValues);

//Color.RGBtoHSB(color.getRed(), color.getGreen(), color.getGreen(), hsbValues);

    hue = hsbValues[HUE];
    //hue = hsbValues[1];
    return hue;
}
```

Line 137 - Change one of the if parameters to look for a length of 5 instead of 6



```
public static Color hex2Rgb(String colorStr) {
    ///*
    if (colorStr.charAt(0) == '#') {
        String str = colorStr.substring(1);
        if (str.matches(HEXADECIMAL_DICTIONNARY)
                && str.length() == RGB_HEXA_LENGTH) {
                //&& str.length() == 5) {
            return getNewColor(str);
        } else if (str.matches(HEXADECIMAL_DICTIONNARY)
                && str.length() == RGB_SHORT_HEXA_LENGTH) {
            return getNewColorShortHexa(str);
```

Line 147 - Change an if parameter to not accept a hex value starting with a #



```
public static Color hex2Rgb(String colorStr) {
    ///*
    if (colorStr.charAt(0) == '#') {
        String str = colorStr.substring(1);
        if (str.matches(HEXADECIMAL_DICTIONNARY)
                && str.length() == RGB_HEXA_LENGTH) {
                //&& str.length() == 5) {
            return getNewColor(str);
        } else if (str.matches(HEXADECIMAL_DICTIONNARY)
                && str.length() == RGB_SHORT_HEXA_LENGTH) {
            return getNewColorShortHexa(str);
        }
    }
    //*/
    //Remove the two // before each block comment indicator in the above portion of the if statement and delete the "else" from the line below this in order
    //to introduce a fault where the method no longer accepts inputs containing a # before the hexstring
    else if (colorStr.matches(HEXADECIMAL_DICTIONNARY)) {
        if (colorStr.length() == RGB_HEXA_LENGTH) {
            return getNewColor(colorStr);
        } else if (colorStr.length() == RGB_SHORT_HEXA_LENGTH) {
            return getNewColorShortHexa(colorStr);
        }
    }
    return null;
}
```

Chapter 6

Our Experiences

Throughout this entire process, our team was largely unaware of the efforts that it took to produce a detailed testing framework. None of us had ever worked on an open source project before and none of us had any testing experience prior to this semester. Even the basics of cloning a repository and extracting methods from the code base was a task that took repetition to learn. This class has given us a more realistic view into how working with a company and collaborating with tens to hundreds of other developers would look like in regards to sharing resources and code on a repository like Github. I believe the one characteristic that we brought into this project that helped us the most was our ability to communicate effectively and efficiently as a team. At each stage of the project, we had planned meet ups inside of class and outside of class and each time everyone was present and attentive. We delegated responsibilities appropriately and had great success in giving individuals tasks that they desired to work on. At no step of the way did we run into any major snags or setbacks that caused us large amounts of stress and it was truly a pleasurable experience working with a team that had an uplifting, positive attitude during every encounter. If there were two major takeaways from this assignment it would be that Google can answer all of your questions and a team that coalesces together can accomplish great goals even when no one has prior experience.

Assignment Critiques

There was not much confusion when it came to doing each part of the group projects. The process seemed very streamlined and clear as to what the goal of each assignment was and how those goals might be achieved. There was open-ended suggestions on how to complete each task while giving enough leeway to formulate a unique solution to the problem. The only confusion that comes to mind was during the start of the class when we picked what open source project we'd be working on. There were quite a few projects that weren't being supported or worked on anymore on the H/FOSS project page. I don't know who would clear that up but that was the only issue that we came across. This process of developing a testing framework for an open source project seems to be a proven method that's be tested through the class many times and the whole process was very clear.