Chapter 2

Experiences

We struggled with deliverable 2 for several reasons. We still feel a little unsure about the Test Template as different people on our teams had varied interpretations of what that meant. We were initially searching for a class that had at least five testable methods, which we never found, but we were informed that the methods didn't have to all come from the same file. We finally chose a file to at least base the first five test cases on, but we didn't think through needing a method that takes input. To find methods to test, we chose ColorCombinaisonImpl.java and decided on the method public boolean equals(Object obj).

Update:
Since the date of deliverable 2 being due, we were informed that the method we chose was not appropriate for testing since it is a Java method, not specific to the Tanaguru's Contrast Finder. Additionally, we have since needed to update our testTemplate.txt file for ease of use with our script.

Report

Once we chose ColorCombinaisionImpl and equals(obj), it became clear on what we need to test. This method checks that the color combination that is being suggested is valid. It will fail if the object is null, isn't a ColorCombinaisionImpl object, or the colors differ between objects. The method will otherwise pass.

Our five tests on this method should test the three ways to return false and cases to return true. Case 1 includes the comparison object being tested against itself, which should return true. Case 2 will be the correct object type with the same values as the comparison object, which we expect to return true. Case 3 will be the correct object, but with color values that differ from the comparison object, which will return false. Case 4 will pass null into the method, which will return false. Case 5 will return false, since the input won't even be the correct object type.

Update:

For the new first five test cases, we chose the method Rgb2Hex, which takes in a Color, meant to be defined by rgb values and returns the hex value of the color. The first test case, takes a random rgb value and returns the correct hex value. The second and third cases test the ends of the color values, testing black with an rgb value of 0,0,0 and a hex value of 000000 and white with an rgb value of 255,255,255 and a hex value of FFFFFF respectively. The fourth case tests invalid integers out of range of rgb values and will return an error. The fifth case tests strings as an invalid input and will also return an error.

Updated Test Plan

The Testing Process

We will have five test cases per method and we will have five methods to test.

Requirements Traceability

Test Case1: The method will take a valid color defined by rgb and change it to hex

Test Case2: The rgb value equating to black will output a hex value with all zeros.

TestCase3: The rgb value equating to white will output a hex value with all Fs.

TestCase4: The method will not allow rgb values out of range.

TestCase5: This method will not accept strings as valid arguments.

Tested Items

ColorConverter.java

Method: rgb2Hex(Color color)

Testing Schedule

10/01/19 - Design 5 test cases on one method

10/18/19 - Create drivers for first 5 test cases and design 10 more test cases

10/25/19 - Create drivers for test cases 6-15 and design 5 more test cases

10/29/19 - Design and build test case automation

11/01/19 - Create drivers for test cases 16 - 20 and design 5 more test cases

11/08/19 - Create drivers for test cases 20 - 25

11/12/19 - Design 20 additional test cases

11/19/19 - Design and inject 5 faults into code so that at least 5 test cases fail

Run all tests as they are created and run previous tests.

Test Recording Procedures

The output of the tests will be written to a file that we will use diff against testCaseOracle files to determine if the tests pass or fail.

Constraints

Time is the biggest constraint working against us. Between classes and work, finding time to work together can be difficult.

Updated Test Cases:

Test Template:

Test ID ## <-- read by script

Requirement being tested

Reserved for comments

package.path.to.Component.method()

DriverName <-- read by script

Test input(s) including command line argument(s) <-- read by script

Expected outputs <-- read by script

Case 1:

01

The method will take a valid color defined by rgb and change it to hex

org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()

Rgb2HexDriver

220 136 15

#DC880F

Case 2:

02

The rgb value equating to black will output a hex value with all zeros.

org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()

Rgb2HexDriver

0 0 0

#000000

Case 3:

03

The rgb value equating to white will output a hex value with all Fs.

org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()

Rgb2HexDriver

255 255 255

#FFFFFF

Case 4:

04

The method will not allow rgb values out of range.

org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()

Rgb2HexDriver

300 -15 2

Exception

Case 5:

05

This method will not accept strings as valid arguments.

org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()

Rgb2HexDriver

"cat" "meow" "dog"

Exception

Test Plan

The Testing Process

We will have five test cases per method and we will have five methods to test.

Requirements Traceability

Test Case1: The object constructed from itself will equal an input of itself, returning true.

Test Case2: The input of the ColorCombinaisonImpl(with same parameters as self) will return true.

TestCase3: The input of the ColorCombinaisonImpl(with different parameters as self) will return false.

TestCase4: The input null will return false.

TestCase5: Input of an object other than ColorCombinaisonImpl (String in this test case) will return false.

Tested Items

ColorCombinaisonImpl.java

Method: equals(Object obj)

Testing Schedule

10/01/19 - Design 5 test cases on one method

10/18/19 - Create drivers for first 5 test cases and design 10 more test cases

10/25/19 - Create drivers for test cases 6-15 and design 5 more test cases

10/29/19 - Design and build test case automation

11/01/19 - Create drivers for test cases 16 - 20 and design 5 more test cases

11/08/19 - Create drivers for test cases 20 - 25

11/12/19 - Design 20 additional test cases

11/19/19 - Design and inject 5 faults into code so that at least 5 test cases fail

Run all tests as they are created and run previous tests.

Test Recording Procedures

The output of the tests will be written to a file that we will use diff against testCaseOracle files to determine if the tests pass or fail.

Constraints

Time is the biggest constraint working against us. Between classes and work, finding time to work together can be difficult.

Test Cases

In these test cases, the comparison object will be constructed as ColorCombinaisonImpl(Color(0,0,255), Color(255,0,0), 50).

Test Template:

1. Test ID

2. Requirement being tested

3. Component being tested

4. Method being tested

5. Test input(s) including command line argument(s)

6. Expected outputs

Case 1:

1. 01

2. The object itself, equals itself.

3. ColorCombinaisonImpl

4. equals(obj)

5. this

6. true

Case 2:

1. 02

2. The object must be of type ColorCombinaisonImpl and valid colors.

3. ColorCombinaisonImpl

4. equals(obj)

5. ColorCombinaisonImpl(Color(0,0,255), Color(255,0,0), 50)

6. true

Case 3:

1. 03

2. The object must be of type ColorCombinaisonImpl and valid colors.

3. ColorCombinaisonImpl

4. equals(obj)

5. ColorCombinaisonImpl(Color(255,0,0), Color(0,0,255), 50)

6. false

Case 4:

1. 04

2. Null is not a valid object.

3. ColorCombinaisonImpl

4. equals(obj)

5. null

6. false

Case 5:

1. 05

2. Only the ColorCombinaisonImpl object is valid

3. ColorCombinaisonImpl

4. equals(obj)

5. String "test"

6. false