Software Engineering Project
Test Plan

Version 1.1
10/19/2017

1. Introduction

Purpose:

The Tanaguru Contrast-Finder module evaluates if the color used by web developers complies with the international regulation established by the WCAG (Web Content Accessibility Guidelines).
This test plan details our testing approach and framework that will execute the testing of Tanaguru Contrast-Finder. Contrast-Finder module is a tool that provides web developers the ability to review and evaluate the colors of a page or an entire web application. Contrast-Finder is reliable, intuitive and self-accessible.

Scope:
This test plan is defined to test methods included in the Contrast-Finder module.

Objective:
The objective of the test plan is to verify that the functionality of Contrast-Finder meets the specifications found on the Tanaguru GitHub documentation

2. Testing Process

The testing process will begin with the selection of methods to be tested.
Only Unit Testing will be performed.

The first step is to design test cases and store them in the folder testCases. The test cases also contain the test data and expected outputs, that will be used to test the methods.

A script “runAllTests. sh” is created to execute multiple test cases.

As tests are executed the results are evaluated as Pass/ Fail in the test report.

If a test is marked as Fail, either a failure in the method was detected or the test itself may have a failure. Both method and test case need to be reviewed.

3. Requirements Traceability

Tanaguru Contrast Finder is a tool which computes the contrast between two colors (background, foreground) and checks if the contrast is valid. Foreground is the text color and background is the page color.
Requirements:

- Both fields can be filled with a hexadecimal value between #000000 and #FFFFFF.
- Note that #ABC works also, It is automatically expanded in the form #AABBCC.
- Both fields can be filled without the '#'character, so FFF or FFFFFF work perfectly.
- The colors ratio between foreground and background has three possible values 3, 4.5 and 7.

4. Tested Items

Method getContrastRatio from ContrastChecker class.
Method getLuminisity from ContrastChecker class.
Method ValidateColor from ColorModelValidator class.
Method hex2Rgb from ColorConverter class.

5. Testing Schedule

Activity	Start	End	Team Member
Prepare Test Plan	September 26	October 3	Fabiola & Luke
Review Test Plan	October 3	October 24	Fabiola & James
Prepare Test Cases	September 26	November 14	All team members
Prepare Test Scripts	October 3	October 18	Julian
Execute 5 Test Cases	October 18	October 31	Julian
Review Testing	October 31	November 14	All team members

6. Test Recording Procedures

1. First, the driver traverses the test cases folder and receives a list of files (testcase#.txt)
2. Once it receives the files, it reads each individual test case and organizes the information in this type of format
    1. testCaseID
    2. Method
    3. Description
    4. fgColor
    5. bgColor
    6. expectedOutput

    By setting up this format, we are able to retrieve each piece of information that we need for executing the driver
3. Next, we format each piece of information into their appropriate data type

4. Methods are executed to retrieve the actual result returned by the method

5. The actual value is compared to the expected value (as specified in test case text file) to produce a pass or fail result

6. Lastly, we format and write the results into a table for proper visual clarity.

7. Software and Hardware requirements

- Java
- Linux (Ubuntu 64-bit)
- Firefox (to display test report)

8. Constrains

- Besides daily production constraints, our only main team constraints have involved timing issues.

9. Tests Cases

Test 01

Requirement being tested: Compute the contrast ratio between 2 equal colors .
Component being tested: ContrastChecker.java
Method being tested: getContrastRatio(String fgColor, String bgColor)
Test input : 255, 255, 255
255, 255, 255
Expected outcome: 1.0

Test 02

Requirement being tested: Compute the contrast ratio between colors white and black.
Component being tested: ContrastChecker.java
Method being tested: getContrastRatio()
Test input: 255, 255, 255
0, 0, 0
Expected outcome: 21.0

Test 03

Requirement being tested: Compute the contrast ratio between 2 colors
Component being tested: ContrastChecker.java
Method being tested: getContrastRatio()

Test input: 0, 0, 0
255, 255, 255
Expected outcome: 21.0

Test 04

Requirement being tested: Gets luminosity as decimal
Component being tested:ContrastChecker.java
Method being tested: getLuminosity(Color)
Test input: 255, 255, 255
Expected outcome: 1.0

Test 05

Requirement being tested
Component being tested: ContrastChecker.java
Method being tested: getLuminosity(Color)
Test input: 0, 0, 0
Expected outcome: 0.0

Test 06

Requirement being tested: String input is hexadecimal value between #000000 and #FFFFFF
Component being tested: ColorModelValidator.java (contrast-finder-webapp/src/java/org/opens)
Method being tested: ValidateColor(String, String, Errors)
Test input: “#FFFFFF”
“#FFFFFF”
Expected outcome: (255,255,255)

Test 07

Requirement being tested: String input is hexadecimal value between #000000 and #FFFFFF
Component being tested: ColorModelValidator.java (contrast-finder-webapp/src/java/org/opens)
Method being tested: ValidateColor(String)
Test input: “FFF”
“ “
Expected outcome: error = “NOT_A_VALID_COLOR”

Test 08

Requirement being tested: String input is hexadecimal value between #000000 and #FFFFFF
Component being tested: ColorModelValidator.java (contrast-finder-webapp/src/java/org/opens)
Method being tested: ValidateColor(String)
Test input: null

null
Expected outcome: error = “NOT_A_VALID_COLOR”

Test 09
Requirement being tested: String input is hexadecimal value between #000000 and #FFFFFF
Component being tested: ColorModelValidator.java (contrast-finder-webapp/src/java/org/opens)
Method being tested: ValidateColor(String)
Test input: “C0C0C0”
“C0C0C0”
Expected outcome: (192,192,192)

Test 10
Requirement being tested: String input is hexadecimal value between #000000 and #FFFFFF
Component being tested: ColorModelValidator.java (contrast-finder-webapp/src/java/org/opens)
Method being tested: ValidateColor(String)
Test input: “#00g”
“#00g”
Expected outcome: error = “NOT_A_VALID_COLOR”

Test 11
Requirement being tested: Return The RGB color from hex Color
Component being tested: ColorConverter (Contrast-finder-utils)
Method being tested: hex2Rgb(String)
Test input: “#000000”
“#000000”
Expected outcome:(0,0,0)

Test 12
Requirement being tested:Return The RGB color from hex Color
Component being tested:ColorConverter
Method being tested: hex2Rgb(String)
Test input: “#FFFFFF”
Expected outcome: (255, 255, 255)

Test 13
Requirement being tested: Return The RGB color from hex Color
Component being tested: ColorConverter
Method being tested: hex2Rgb(String)

Test input:”#000F”
Expected outcome: null

Test 14

Requirement being tested: Return The RGB color from hex Color
Component being tested: ColorConverter
Method being tested: hex2Rgb(String)
Test input: “#00000G”
Expected outcome: null

Test 15

Requirement being tested: Return The RGB color from hex Color
Component being tested: ColorConverter
Method being tested: hex2Rgb(String)
Test input: “#FF0”
Expected outcome: (255.255,0)