Testing and Integration Standards Document

Roger Tan

Document Control

Version Number	Modified By	Date	Section(s) Modified	Comments
1.0	R. Tan	02/02/2014	All	*Testing and Integration Standard Document.*
1.1	R. Tan	27/02/2014	2.3/5 & 6	*Added Manual Testing procedure and template.*
1.2	R. Tan	03/06/2014	All	*Using the Document Template specified in QA manual.*

Contents

1 Introduction

This document provides detailed information on the format and structure of the testing procedure.

Any queries about the Testing Standards Document should be referred to the Testing and Integration manager.

2 Testing

2.1 Naming Convention

When comes to writing any test method, one must adhere closely the Standard Java Naming Convention as stated in the Coding Standards Document. This is required so as to produce readable and consistent tests.

2.2 Automated Testing

- A test for each method should be derived from the design requirements.
- Only test for one concept per test.
- Minimum test should be done to satisfy the user story.
- Refactoring the tests after passing the acceptance test.

2.3 Manual Testing

- Visual inspection to see if the codes have performed what it should. (e.g. when the PLAY button is pressed, the video starts playing.)
- To check for bugs if multiple buttons or features are executed at the same time.

3 Comments

3.1 Header Comments

At the beginning of test class, include the following information:

Programmer's name

Date created

Description of the test class

4 Test Structure

An example test structure:

```
public class AccountManagerTest {
private AccountManager manager;
private StudentAccount account;
        @Before
        public void setUp() throws Exception {
                manager = new AccountManager();
                manager.addAccount("Joe Bloggs");
                manager.addAccount("Jonas Johnson");
        }
        @Test
        public void createMultipleAccounts() throws Exception {
                account = manager.getAccount("Joe Bloggs");
                assertEquals("Joe Bloggs", account.getAccountID());
                account = manager.getAccount("Jonas Johnson");
                assertEquals("Jonas Johnson", account.getAccountID());
        }
}
```

5 Manual Testing Report Layout

- Create a new Class document and name it the Class that you are testing with an additional word "Test" behind. For example ImageDisplayTest, ExitSlideshowTest and etc.
- Next, remove the Class constructor inside and use the following template shown below to record the manual test results

```
/*
 Programmers: Roger & Prakruti
 Date created: 27/2/2014
 Description: Exit Slideshow to MainMenu Test

 Program Description: This program is designed to create a slide from the main
                      menu.
The program is capable of exiting the slide and return to the main menu.

Program Setup: This program is designed to create a fullscreen slide whenever
the "Create Slide" button is being pressed. In the slide, there's a "Exit Slide"
button to close the slide and return to main menu. This could also be done by
pressing the "ESC" key.

Test Results: When "Create Slide" button is pressed, a separate fullscreen window
(slide) pops up.
               When "Exit Slide" button is pressed, the slide window closes and
return to main menu.
              When "ESC" key is pressed, the slide window closes and return to
main menu.
 */
```

6 Combination of Automated and Manual Testing

There will be cases where combination of both automated and manual testing will take place. Hence in such cases, have them both in a same test class and place the manual test after the automated test.

7 Debugging

If any of the tests keeps failing, the team responsible for coming up with the test method for that particular component will first have to ensure that the test method is appropriately implemented. After ensuring that the test method is properly implemented, the team has to report to the Testing and Integration manager and the manager will then inform the Software manager that there is a problem with the code provided.

8 Integration

When comes to integrating several different modules together, a specified set of test methods is used to ensure that the fundamental features of the product are still functioning accordingly. This is done so as to check and ensure that the integration process does not break any of the fundamental features.

However, after any integration process, the specified set of test methods tends to fail the acceptance test, it means that something has broken while integration that certain modules together. The team responsible for that integration will then have to inform the Testing and Integration manager so as to inform the Software manager to have their team to look into that issue.

For a step by step integration procedure, please refer to the Integration Guidelines document.