Software Testing Document

for

TalkBox

Group 15

York University
EECS2311 Software Development Project

04. February 2019

Contents

Chapter 1

Introduction

1.1 Purpose

This document provides information on test cases for the TalkBox application. This document covers the several test cases the application has, as well as justified derivation of each of these test cases. Test case derivation is justified thoroughly for both the simulator component of the TalkBox, as well as the configuration application component. The sufficiency of each test case is provided through this document as well as test coverage.

1.2 Testing Checklist

There are two testing checklists in the Testing Checklist chapter, one for the TalkBox configuration application and another for the TalkBox simulator application. These checklists are broken into two subsections of the chapter.

1.3 Testing Case Derivation

In the Test Case Derivation chapter of this document, a through description of how each testcase was derived is provided. There are two subsections to the Test Case Derivation chapter, one for the configuration application and another for the simulator application. In each of these subsections the respective applications test case derivation is provided.

1.4 Testing Case Sufficiency

In the Test Case Sufficiency chapter of this document, a justification for each test case from the Test Case Derivation chapter is provided. There are two subsections to the Test Case Sufficiency chapter, one for the configuration application and another for the simulator application. In each of these subsections the respective applications test cases are justified with proven sufficiency

1.5 Testing Case Implementation

In the Test Case Implementation section of this document, a description of how each test case is implemented is provided. There are two subsections to the Test Case Implementation chapter, one for the configuration application and another for the simulator application. In each of these subsections, the respective applications test case implementations are shown and justified.

1.6 Test Coverage

In the Test Coverage chapter of this document the entire Test Coverage will be shown after running all tests. Individual unit tests will also have their Test Coverages shown per test class.

Chapter 2

Overall Description

2.1 Testing Documents Description

Throughout this document there are several chapters that show different test cases as well as the respective derivation, sufficiency, and implementation of each of these test cases. A test coverage for each test as well as all tests is also provided in this document. An understanding of the testing done and developed during the making of the TalkBox is thoroughly analyzed and conveyed throughout this document.

Chapter 3

Testing Checklist

3.1 TalkBox Configuration Application

Recorder Class Tests:

Tests	Pass / Fail	Comments
setup()	pass	Setting up objects and fields.
testInitialFields()	pass	Testing the initial fields of recording class
testRecording()	pass	Testing recording an audio file using the system and saving it to a directory
testButtons()		Testing the feature of changing the amount of buttons of the application

Simulator Preview (SimPreview) Class Tests:

Tests	Pass / Fail	Comments
setup()	pass	Setting up objects and fields.
testingUpdatingButtons()	pass	Testing update button method for the sim preview
testPlayingSound ()	pass	Testing audio output of application audio files
TestErrorPlayinSound()	Pass	Testing failure to play audio file due to wrong file or mic not found
testPlayinMissingSoundFile()	Pass	Testing playing audio if file is missing

PlayEditToggle Class Tests:

Tests	Pass / Fail	Comments
setup()	pass	Setting up objects and fields.
testAddingAudioToButtons()	pass	Testing adding audio to buttons in “Edit Mode”
testChangingButtonLabel()	pass	Testing changing the name of a button using naming feature in “Edit Mode”

ProfilesPanelTest Class Tests:

Tests	Pass / Fail	Comments
setup()	pass	Setting up objects and fields.
test()	Pass	This test allows several features of the profiles panel to be tested, including creating new profiles, and accessing profiles that do not exist.

TalkBoxConfig Class Tests:

Tests	Pass / Fail	Comments
setup()	pass	Setting up objects and fields.
testSetNumAudioButtons()	pass	Testing update number of audio buttons on the TalkBox
testInitialFields	pass	Testing the intial set fields of the TalkBoxConfig app. (ex. Number of buttons)
TestJPaneSplits	Pass	Testing the different instances that are created by the configApp through JSplitPane components

3.2 TalkBox Simulator Application

TalkBox Simulator Tests (TalkBoxSimTest):

Tests	Pass / Fail	Comments
setup()	Pass	Setting up objects and fields.
updateButtonsNumber()	Pass	Testing updating number of buttons method
testPlayingSound()	Pass	Testing playing sound from button one.

Chapter 4

Test Case Derivation

4.1 How Test Cases Were Derived: TalkBox Configuration Application

RecorderTest:

The Recorder tests cases are specifically tailored toward this class and its features. The test case for the Recorder class is derived by accounting for all vital features that are associated with the recorder class, including any errors that may result in the class from a user's usage. For example, testing recording audio is a key feature of this class, and thus a test method is implemented for it. A further break down of each test method is given in Chapter 5 and Chapter 6, showing exactly which tests are done within this test case.

SimPreviewTest:

The SimPreview tests cases are specifically tailored toward this class and its features. The test case for the simPreview class is derived by accounting for all vital features that are associated with the functionality of the simPreview class. For example, playing audio from a button is a vital feature of this class and thus it has a test method implemented for this feature. A further break down of each test method is given in Chapter 5 and Chapter 6, showing exactly which tests are done within this test case.

PlayEditToggleTest:

The PlayEditToggle tests cases are specifically tailored toward this class and its features. The test case for the PlayEditToggle class is derived by accounting for all necessary features of this class. For example, switching into 'Edit Mode' is a key feature of this class, and this a test method is implemented to test this feature. A further break down of each test method is given in Chapter 5 and Chapter 6, showing exactly which tests are done within this test case.

ProfilesPanelTest:

The ProfilesPanel tests cases are specifically tailored toward this class and its features. The test case for the ProfilesPanel class is derived by accounting for all the vital features of this class. For example, creating and loading a new profile is a key feature of this class, and thus the respective test method is derived and implemented. A further break down of each test method is given in Chapter 5 and Chapter 6, showing exactly which tests are done within this test case.

TalkBoxConfigTest:

The TalkBoxConfig tests cases are specifically tailored toward this class and its features. The test case for the TalkBoxConfig class is derived by taking into consideration all the key features that are rooted from this class. For example, the initial fields of the configuration are tested upon launch to ensure the application will correctly launch. A further break down of each test method is given in Chapter 5 and Chapter 6, showing exactly which tests are done within this test case.

4.2 How Test Cases Were Derived: TalkBox Simulator Application

TalkBoxSimTest:

The TalkBoxSim tests cases are specifically tailored toward this class and its features. The test case for the TalkBoxSim class is derived by taking into consideration all key features that are associated with the TalkBoxSim class. For example, testing the playback of audio from the simulator is a key feature of the TalkBox Simulator application, and thus a test method is implemented respectively. A further break down of each test method is given in Chapter 5 and Chapter 6, showing exactly which tests are done within this test case.

Chapter 5

Test Case Implementation

5.1 How Are Test Cases Implemented

The implementation of each of the test cases is shown below by breaking down all test methods within the test case.

RecorderTest:

testInitialFields(): The initial fields of the recorder class are tested as upon launch they should have a certain state. For example, isRecording field should be false when the application is first launched to ensure there is no recording in progress.

testRecording(): The user is able to record from the recording panel, and this method tests the applications recording functionality.

testButtons(): The user is able to change the number of buttons from the recorder panel, and this method allows a test to see if this functionality is working correctly, by preforming a change on the amount of buttons.

SimPreviewTest:

setup(): A configurator object is created and used to create a simulator preview object. These objects are needed to test the PlayEditToggle class.

testingUpdatingButtons(): The user is able to change the number of buttons from the recorder panel, and this method allows a test to see if this functionality is working correctly, by preforming a change on the number of buttons. This updates the number of buttons on the simulator preview.

testPlayingSound(): This method tests playing sound from a button, which is an essential feature of the simulator preview as well as the simulator in general.

testErrorPlayingSound(): This method tests what happens if there is an error playing back audio, for example if a file is null, the correct handled.

testPlayingMissingSoundFile(): This method tests what happens if there is an error playing back audio due to a missing audio file.

PlayEditToggleTest:

setUp(): A configurator object is created and used to create a simulator preview object and recorder object. These objects are needed to test the PlayEditToggle class.

testChangingButtonLabel(): When in edit mode, the user is able to change the labels on each button. This method tests that functionality using the respective JTextFields and JButtons.

testAddingAudioToButtons(): When in edit mode, the user is able to add audio to a button of their choice. This method allows a testing of this feature by using the respective JTextFields and JButtons.

ProfilesPanelTest:

setup(): This method creates a TalkBoxConfig object as well as a profiles panel object, both of which are used to test the ProfilesPanel class.

test(): This method first test the initial fields of the profiles, for example one default profile and hence a size comparison of profiles with 1. The default profile name is also tested. After this, new profiles are created just as they would be by a user by clicking the respective JButtons. The creation of these profiles is then tested to see if it was successful. Accessing a profile index (profile that does now exist), is also tested in this method to ensure the ArrayIndexOutOfBoundsException is handled accordingly.

TalkBoxConfigTest:

setUp(): A configurator object is created. This object is needed to test the main configurator class. A call to the main method of the configuration app is allow done in this method.

testSetNumAudioButtons: This method sets the number of audio buttons for the configuration app and tests if this change took place.

testInitialFields(): This method tests all initial fields of the TalkBoxConfiguration application to ensure they are correctly set upon launch of the application.

testJPaneSplits(): This method tests if all objects created by the TalkBoxConfiguration application are correctly initialized. This intasiation of objects takes place in the controlsProfileSplit class and the SimRecorderSplit class.

TalkBoxSimTest:

setUp(): A configurator object is created as well as a simulator object and buttons panel object. These objects are required in testing the TalkBox Simulator application.

updateButtonsNumber(): The user is able to change the number of buttons on the simulator when they are configurating the application, and this method tests the updateButtons() method in the TalkBoxSimulator class, and then check to see if this change took place accordingly.

testPlayingSound(): This method allows a testing of clicking a button and getting audio playback from that button without an error.

Chapter 6

Test Case Sufficiency

6.1 Why Test Cases Are Sufficient: TalkBox Configuration Application

RecorderTest:

testInitialFields(): This method is sufficient in testing the initial fields as it correctly tests all initial fields of the recorder class.

testRecording(): This method is sufficient in testing the recording feature as it creates a new audio file that is 2 seconds in length and correctly places it in the TalkBoxData directory.

testButtons(): This method is sufficient in testing the change in the amount of buttons, as it will update the number of buttons then check the simulator to see if the number of buttons has actually changed.

SimPreviewTest:

setup(): This method correctly sets up all necessary fields needed to thoroughly test the SimPreview class, and thus it is respectively sufficient.

testingUpdatingButtons(): This method is sufficient in testing the change in the amount of buttons, as it will update the number of buttons then check the simulator to see if the number of buttons has actually changed.

testPlayingSound(): This method is sufficient in testing audio playback from a button as it will correctly play audio from a given button, and test if the audio has been played correctly using a byte output stream.

testErrorPlayingSound(): This method is sufficient in testing the possibility of an error in playing sound due to an incorrect file name, or null file name. It correctly tests the error that is handled during this special case.

testPlayingMissingSoundFile(): This method is sufficient in testing the possibility of a missing audio file that is supposed to be associated with a button. It correctly tests the error that is handled during this special case.

PlayEditToggleTest:

setup(): This method correctly sets up all necessary fields needed to thoroughly test the PlayEditToggle class, and thus it is respectively sufficient.

testChangingButtonLabel(): This method is sufficient as it correctly changes the button label, and then check the simulator preview to ensure that the change has taken place.

testAddingAudioToButtons(): This method is sufficient as it correctly goes through the process of adding an audio recording to a button. It then checks to see if the audio file has correctly been associated with the button, hence this method is sufficient.

ProfilesPanelTest:

setup(): This method correctly sets up all necessary fields needed to thoroughly test the ProfilesPanel class, and thus it is respectively sufficient.

test(): This method is sufficient in testing several aspects of the ProfilesPanel class, as it tests key features of the class. For example, creating new profiles and accessing new profiles is tested thoroughly, including accessing profiles that do not exist. This test method is hence sufficient in testing the current profilesPanel features.

TalkBoxConfigTest:

setUp(): This method correctly sets up all necessary fields needed to thoroughly test the TalkBoxConfig class, and thus it is respectively sufficient.

testSetNumAudioButtons: This method is sufficient as it changes the number of buttons on the simulator preview via the TalkBoxConfig class, and then tests to see if this change took place, hence allowing sufficiency for this method.

testInitialFields(): This method is sufficient in testing the initial fields as it correctly tests all initial fields of the TalkBoxConfig class upon launch of the application.

testJPaneSplits(): This method is sufficient as it correctly creates references to the objects that are created when a TalkBoxConfig. It then tests to see if these references are to the correct objects, hence allowing sufficiency for this testing method.

6.2 Why Test Cases Are Sufficient: TalkBox Simulator Application

TalkBoxSimTest:

setUp(): This method is sufficient in creating the necessary fields required to test the TalkBox Simulator application.

updateButtonsNumber(): This method is sufficient in testing the updating off the number of buttons on the simulator by using the respective updateButtons() method to change the amount of buttons on the simulator. It then tests equality of number of buttons to see if this change took place.

testPlayingSound(): This method is sufficient in testing audio playback from a button as it will correctly play audio from a given button, and test if the audio has been played correctly using a byte output stream.

Chapter 7

Test Coverage

7.1 Test Coverage Metrics

Individual Test Classes

RecorderTest:

TalkBox	Coverage	Covered Instructions	Missed Instructions	Total Instructions
src folder	72.6%	2830	1067	3897

SimPreviewTest:

TalkBox	Coverage	Covered Instructions	Missed Instructions	Total Instructions
src folder	50.8%	1981	1916	3897

PlayEditToggleTest:

TalkBox	Coverage	Covered Instructions	Missed Instructions	Total Instructions
src folder	60.2%	2345	1552	3897

ProfilesPanelTest:

TalkBox	Coverage	Covered Instructions	Missed Instructions	Total Instructions
src folder	50.1%	1953	1944	3897

TalkBoxConfigTest:

TalkBox	Coverage	Covered Instructions	Missed Instructions	Total Instructions
src folder	49.4%	1926	1971	3897

TalkBoxSimTest:

TalkBox	Coverage	Covered Instructions	Missed Instructions	Total Instructions
src folder	62.6%	2439	1458	3897

AllTests:

TalkBox	Coverage	Covered Instructions	Missed Instructions	Total Instructions
Whole Project	82.2%	3677	795	4472