

---

# **TALKBOX TEST PLAN**

---

Version 1.0

04/3/2019

---

# TABLE OF CONTENTS

<b>1 INTRODUCTION.....</b>	<b>3</b>
1.1 Purpose of The Test Plan Document .....	3
1.2 Test Suspension / Resumption Criteria .....	3
<b>2 FUNCTIONAL TESTING .....</b>	<b>3</b>
2.1 Items to be Tested / Not Tested.....	3
2.2 Test Approach(s) .....	4
2.3 How these tests were developed.....	4
2.4 Why these test cases suffice .....	5
2.5 Test Pass / Fail Criteria .....	5
2.6 Test Entry / Exit Criteria .....	5
2.7 Test Suspension / Resumption Criteria .....	5
<b>3 PERFORMANCE TESTING .....</b>	<b>6</b>
3.1 Test Risks / Issues .....	6
3.2 Items to be Tested / Not Tested.....	6
3.3 Test Approach(s) .....	6
3.4 How these tests were derived .....	6
3.5 Why these tests are sufficient.....	7
3.6 Test Pass / Fail Criteria .....	7
3.7 Test Entry / Exit Criteria .....	7
3.8 Test Suspension / Resumption Criteria .....	7
<b>4 JUNIT TESTING .....</b>	<b>8</b>
It was determined that currently only function and performance testing was required, however more tests are going to be implemented in the future. 오류!      책갈피가 정의되어 있지 않습니다.	
<b>6 LIST OF ISSUES .....</b>	<b>9</b>
<b>7 COVERAGE.....</b>	<b>9</b>
<b>8 APPENDIX A: REFERENCES.....</b>	<b>10</b>

note that while all responsibility of testing is noted under peter, all group members participated in testing during the development of any feature.

## 1 INTRODUCTION

### 1.1 PURPOSE OF THE TEST PLAN DOCUMENT

The Test Plan document documents and tracks the necessary information required to effectively define the approach to be used in the testing of the project's product. The Test Plan document is created during the Planning Phase of the project. Its intended audience is the project manager, project team, and testing team. The Test Plan document is created to ensure that there are little to no issues with any software that is released for consumption.

### 1.2 TEST SUSPENSION / RESUMPTION CRITERIA

If a problem or error was found that was vital to operation of the system and the various tests that were being performed, all tests were put to an immediate halt until such issues were resolved. Smaller problems such as an incorrect implementation of a method that did not affect the entire system's performance did not halt the testing process but were instead noted and to be fixed later.

## 2 FUNCTIONAL TESTING

### 2.1 ITEMS TO BE TESTED / NOT TESTED

Item to Test	Test Description	Test Date	Responsibility
GUI	Ensuring the interface worked on monitors of various sizes and resolutions. Testing all the various buttons and their functions, ensuring they work as intended.	2/24/2019-4/3/2019	Peter
Voice Recording	Testing the voice recording feature with several different microphones and on different devices. Also used various volume ranges and recording lengths.	2/24/2019-4/3/2019	Peter
Configuration	Tested the configuration of the buttons and button sets, adding and removing buttons, changing button images, saving and creating tbc files, as well as finding implementation errors that can be abused.	2/24/2019-4/3/2019	Peter
Sound Playback	Ensure all buttons produced the intended sound.	2/24/2019-4/3/2019	Peter

## **2.2 TEST APPROACH(S)**

The approach used to test these various items was to go use every feature of the program in every way currently possible. Every test was repeated multiple times and some tests were run continuously to ensure that no problems occurred only after a certain timeframe had past. Several people were also invited to come test the software and were asked to note any issues and push the program to its limits.

## **2.3 HOW THESE TESTS WERE DEVELOPED**

In order to ensure that the program works as intended we must extensively test every possible function and every possible mis-use of the program. Only after doing so would we be confident that our software was ready for release.

We first looked at our Requirement Document to find all the intended features of the program:

- User can add/remove audio buttons from current sets.
- User can add/remove audio sets.
- User can record audio files to be assigned to a button.
- User can name recorded audio file.
- User can choose buttons from list of available audio files.
- User can swap to any audio set at any time.
- User can rename audio sets.
- User can give each button a unique picture, searching their device for such pictures.
- User can swap current button with another.
- User can save changed setting to current TBC file.
- User can save current settings as a new TBC file.
- User can return to TalkBox App at any time, discarding unsaved changes.
- A physical device integrated with an application using Java. The device must have buttons that when pressed provides a set audio output associated with that button.
- Allowing user to switch between multiple (possibly infinite) amounts of button sets, each with different audio outputs.
- Can stop the audio playback anytime during the playback.
- Interface is easy to use and intuitive.
- Can jump to any audio set at any time.
- Can enter the configuration settings easily.
- Can switch between configuration settings.

We derived the majority of our testing based on these intended functions and ensuring that they all worked properly and efficiently. We found that all the functions fell under four main categories: GUI, functions that were focused on what was displayed to the user and how they could interact with the program. Voice Recording, a major project requirement that allows the users or configure and create their own audio files. Sound Playback, the sounds that play after a sound button is pressed. And Configuration, being able to change and save multiple setting to fit the caretaker and user's needs.

## **2.4 WHY THESE TEST CASES SUFFICE**

We concluded that if our program was able to accomplish all of the features in our requirements document, and do so in a timely and efficient manner, then we would have enough confidence to say that that all of the program's functionality testing was sufficient. We went beyond this however and also tried to ensure that there were no other unintended uses of the program, and if any testing revealed such functions, they were promptly removed.

## **2.5 TEST PASS / FAIL CRITERIA**

A successful test was defined by no issues being found and the system working as intended. A failed test was one that determined that something needed to be changed to provide a different result/implementation than the system in its current state.

## **2.6 TEST ENTRY / EXIT CRITERIA**

When a feature was believed to have been completed it entered the function testing phase. After a feature had been tested thoroughly and all issues having been resolved, it exited this testing phase until another change was made or until the project reached completion.

## **2.7 TEST SUSPENSION / RESUMPTION CRITERIA**

When alarming issues are found with the program, testing is halted, and the issue is addressed. These issues include ones that broke what the entire system was intended to do. If the issue was fixed and no immediate occurrence of the problem was visible, testing resumed.

### 3 PERFORMANCE TESTING

#### 3.1 TEST RISKS / ISSUES

Current issues with the performance of the system are that it cannot be tested on lower end devices. We currently do not know the minimum requirements to run the program as intended. This is not expected to be a large problem as the program is currently not very resource intensive, however attempts to find these limitations will be performed in the future.

#### 3.2 ITEMS TO BE TESTED / NOT TESTED

Item to Test	Test Description	Test Date	Responsibility
GUI	Ensure that the GUI works smoothly. We do so by clicking buttons in rapid succession and try to overload the program.	2/24/2019-4/3/2019	Peter
Configuration	Ensure that the configuration menu works smoothly. As with the GUI, buttons are pressed in rapid succession.	2/24/2019-4/3/2019	Peter
Voice recording	Ensure that the quality of the sound recorded is acceptable given the microphone used. Recording of different lengths and volumes were made with several microphones and on several devices.	2/24/2019-4/3/2019	Peter
Sound playback	Ensure that all sound is played smoothly and is not choppy given a microphone of decent quality.	2/24/2019-4/3/2019	Peter

#### 3.3 TEST APPROACH(S)

Buttons were pressed in quick succession and the program was run in ways that were intended to crash, lag, or break the program. These included providing inputs that were unexpected or overloading the application with too many instructions at once. Tests were also performed when a device had many copies of the TalkBox App open simultaneously in a test to further push the application's boundaries.

#### 3.4 HOW THESE TESTS WERE DERIVED

In trying to find the performance flaws of the application we found that it was necessary to push the system to its limits and beyond. If we can create repeatable scenarios in which the performance of the program suffered we defined that to be a successful way of finding problems with the software. Since we had already tested all functions of the program with many expected inputs, we decided to test them with various strange or unconventional inputs that the system may not have been coded to handle. If any of these such inputs caused issues, a solution to said input cases were added to the code of the software.

### **3.5 WHY THESE TESTS ARE SUFFICIENT**

In testing the performance of the application, we determined that the only thing that was necessary was if the program could run smoothly regardless of how the user interacts with the interface. If we are able to accomplish that we concluded that that would be a successful test of the system's performance.

### **3.6 TEST PASS / FAIL CRITERIA**

The test was successful if the program ran smoothly and was unsuccessful if there was any lag or crashing of the system.

### **3.7 TEST ENTRY / EXIT CRITERIA**

Once a function/feature had passed the function testing phase it moved onto the performance testing phase. Once all tests were completed and the program was confirmed to run smoothly, it exited performance testing.

### **3.8 TEST SUSPENSION / RESUMPTION CRITERIA**

When the performance of any feature/function of our program is found to be unsatisfactory, development of any features that are based of said function will be halted until the issue with performance can be fixed. After the issue is resolved the development of all features halted due to the issue will continue.

## 4 JUNIT/GRADLE TESTING

A list of the Junit/gradle tests are shown below:

```
test.java.TalkBox.SoundTest > testTrue() PASSED
test.java.TalkBox.SoundTest > test() PASSED
test.java.TalkBox.SoundTest > testExpectedExceptionFail() PASSED
test.java.TalkBox.TalkBoxGuiTest > test() PASSED
test.java.TalkBox.TalkBoxTest > AddAudio_test() PASSED
test.java.TalkBox.TalkBoxTest > setNumberOfAudioSets_test() PASSED
test.java.TalkBox.TalkBoxTest > setAndGetHasAudio_test() PASSED
test.java.TalkBox.TalkBoxTest > RemoveAudio_test2_All() PASSED
test.java.TalkBox.TalkBoxTest > setAudioFileNames_test2() PASSED
test.java.TalkBox.TalkBoxTest > getNumberOfAudioButtons_test() PASSED
test.java.TalkBox.TalkBoxTest > getRelativePathToAudioFiles_test() PASSED
test.java.TalkBox.TalkBoxTest > setAndGetFile_test() PASSED
test.java.TalkBox.TalkBoxTest > setAndGetImages_test() PASSED
test.java.TalkBox.TalkBoxTest > setAndGetSetNames_test() PASSED
test.java.TalkBox.TalkBoxTest > getTotalNumberOfButtons_test() PASSED
test.java.TalkBox.TalkBoxTest > NumberOfAudioSets_increase_test() PASSED
test.java.TalkBox.TalkBoxTest > NumberOfAudioButtons_increase_test() PASSED
test.java.TalkBox.TalkBoxTest > NumberOfAudioSets_decrease_test() PASSED
test.java.TalkBox.TalkBoxTest > setAndGetSettingsList_test() PASSED
test.java.TalkBox.TalkBoxTest > NumberOfAudioButtons_decrease_test() PASSED
test.java.TalkBox.TalkBoxTest > RemoveAudio_test() PASSED
test.java.TalkBox.TalkBoxTest > setAudioFileNames_test() PASSED
```

The majority of these test cases comprised of tests to the TalkBox class, the class that contained the majority of the functions that would be used by other classes to fulfill the required features of the program. We determined that if all of these test cases were working as intended, then any new functions that were based on these would also work as intended. This conclusion was correct as we had no further issues with any of these functions when further developing the application.









## 5 LIST OF ISSUES

Issue with	Issue Description	Date found	State of issue	Solution Description
GUI	Interface is always in front of all other applications.	2/24/2019	FIXED	Changed interface settings to allow it to go behind other programs when not selected.
Configuration settings	Users could create an infinite number of buttons per set.	2/24/2019	FIXED	Limited number of buttons per set to 6
GUI	Buttons with large names do not display properly.	2/24/2019	FIXED	Button size can no longer be changed.
Performance	Clicking buttons in rapid succession causes the program to stutter.	2/24/2019	FIXED	Code was optimized.
GUI	Volume buttons currently do nothing.	2/24/2019	Removed	Volume buttons removed and deemed unnecessary.
Configuration	Users cannot select images outside of the TalkBoxData images folder.	4/3/2019	FIXED	A function was implemented to determine if a selected image was not in the images folder and would duplicate it into the Images folder if it was not already present.

## 6 COVERAGE

It was determined that currently only function and performance testing, and Junit/Gradle testing was required.

After testing all the features of the program, the coverage looked as follows:

>  ConfigurationGUI.java	 95.4 %	5,124	245	5,369
>  TalkBoxGui.java	 94.1 %	2,336	147	2,483
>  RecordGUI.java	 94.6 %	511	29	540

(Note that when only testing specific features/functions the coverage was much lower. This allowed us to determine what code was running as well as to determine what was causing errors and issues.)

## 7 APPENDIX A: REFERENCES

The following table summarizes the documents referenced in this document.

Currently empty but will be kept for future use.

Document Name and Version	Description	Location
<i>Requirements Document 1.0</i>	A document containing all the requirements of the application for the consumer. Includes all functions and acceptance cases of the program.	In the Documentation folder in the program's file location.
<i>User Manual 2.0</i>	A document containing detailed instructions on how to use the TalkBox App and Configuration.	In the Documentation folder in the program's file location.