EECS 2311

Authoring Application

User Manual

Jinho Hwang

Rajvi Chavada

Andrew Maywapersaud

Table of Contents

Pages	Description
4	1.0 Revision History
5	2.0 Program Overview
6-8	3.0 Getting Started
9-12	4.0 Scenario Editor: Load Scenario
13-17	5.0 Scenario Editor: Creating and Editing a Scenario
18-19	6.0 Voice Recorder
20	7.0 Screen Reader Compatibility

List of Figures/Media

Pages	Figure Description
6	Figure 1: Project Imported into Eclipse
7	Figure 2: Launching the Authoring Application
7	Figure 3: Command line launch of Authoring Application
8	Figure 4: Scenario Editor
9	Figure 5: Load Scenario
10	Figure 6: Scenario editor after Load scenario
10	Figure 7: Visual or Audio prompt
11	Figure 8: Simulated braille cells
11	Figure 9: Pin alignment for 1,3 versus 4,6
12	Figure 10: Pins 2,5 versus 1,4
13	Figure 11: Scenario Maker
14	Figure 12: Adding a display string command to new scenario
14	Figure 13: Adding a skip command
15	Figure 14: Moving the skip command downwards
16	Figure 15: Edit Scenario
18	Figure 16: Voice Recorder
19	Figure 17: Creating VoiceTest.wav
19	Figure 18: Load Sound Window
20	Video 1: VoiceOver demo

1.0 Revision History

Date	Revision
02/22/18	Revision 1. Midterm Design and Implementation
02/23/18	Revision 2. Midterm Design- AudioPlayer added

2.0 Program Specification

Welcome to the Authoring Application. This application simulates a treasure box braille with the purpose of allowing the educator to convey text to users. The users may include the visually impaired. With this software, the user can load, create and save scenarios to be presented. Further, two modes are available. If the educator, user, or a subset of the users are not visually impaired, a scenario editor simulates a visual player using audio, interactive buttons and braille cells. In contrast, an audio player mode exists that plays audio files to receive the information through auditory signal. The scenarios involve a set of instructions or passages. These instructions are illustrated through the braille letters or audio. Buttons allow a response to the instructions during interactive modes.

The program was constructed using the Java programming language. Specifically, SceneBuilder was installed on the Eclipse IDE to design the application's user interface using JavaFx libraries. Most of the functionality is implemented with JavaFx. The Swing and Awt library comprise the remaining code. The Eclipse IDE used was the current version: Oxygen 2.0. It is imperative any user of this application is using the latest IDE in particular Junit 5 and Java 8.

It must be noted that this application is still in the beta stage. Thus, this manual will be continuously revised until the finalization of the application. Further, this manual was created using a MacBook. Therefore, the graphics will be iOS related. This should not affect the use of the application.

3.0 Getting Started

The prerequisites to this section are that the readers of this manual have successfully extracted the program. In addition, it is assumed the user has the current version of java installed, the Eclipse IDE equipped with Junit 5 or a terminal that can compile and run java applications.

After importing the project to Eclipse, the following directory structure should be seen:

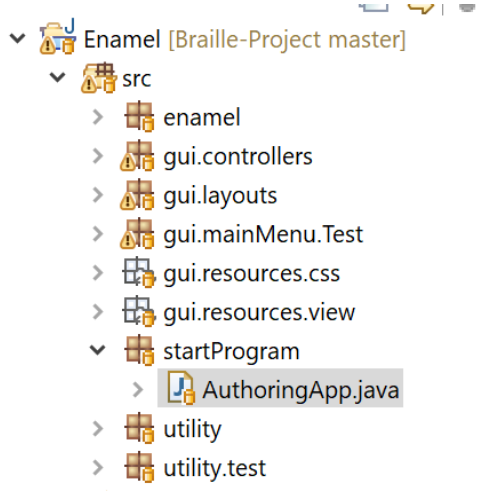

Fig.1 Project Imported into Eclipse

To launch the application, click on the AuthoringApp.java class in the startProgram package and run the program. The following user interface will display (figure.2):

Fig.2 Launching the Authoring Application

Alternatively, the application can be run from the command line using the following commands. Ensure your current directory is correctly set to the startProgram package.

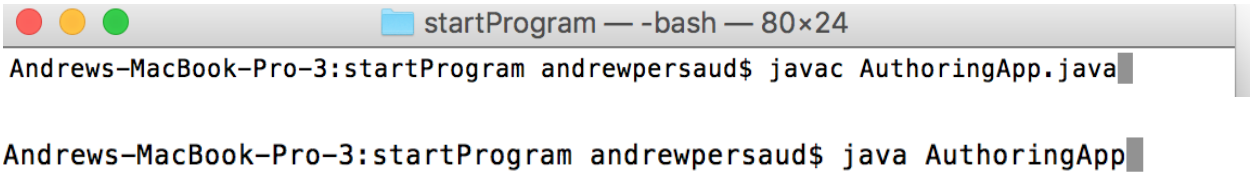

Fig.3 Command line launch of Authoring Application

After confirming this display successfully launched, click on 'Scenario Editor'. We will cover 'Voice Recorder' in a subsequent section. After clicking on 'Scenario Editor', the interface should change to the following window in figure 4.

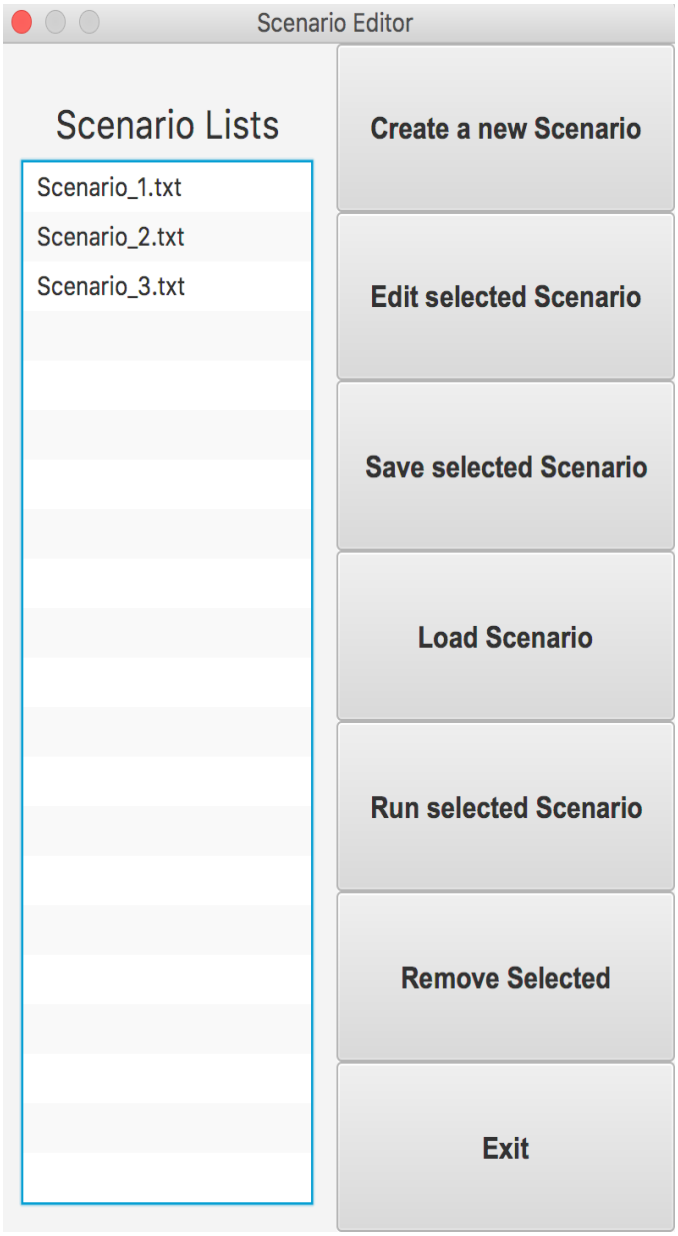

Fig.4 Scenario Editor

This interface will allow the educator to perform the actions specified above. The following section will traverse through these actions.

4.0 Scenario Editor: Load Scenario

The first section we will explore is the 'Load Scenario'. Note that the project will include scenario files for you to load and run to get a perception of the application. It is assumed the reader of this manual is competent in creating their own scenario files to be loaded.

After clicking on 'load scenario', the following window should appear allowing the user to select a file.

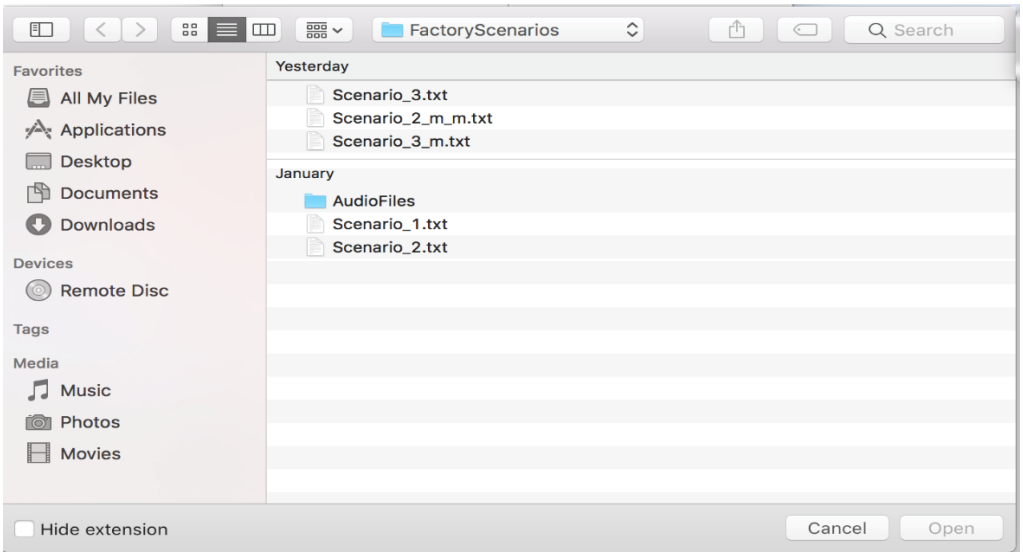

Fig 5. Load Scenario

Your folder may not look exactly as the above, however what should be there are Scenario_1, Scenario_2 and Scenario_3 text files. For convenience, these three text files are pre-loaded.

In the event they do not come pre-loaded, click on Scenario_1 and then 'Open'. Repeat the above for Scenario_2 to ensure that multiple files are shown on the Scenario Editor. After these steps, the following should display in figure 6. Note that all three scenario files will display if they appear pre-loaded.

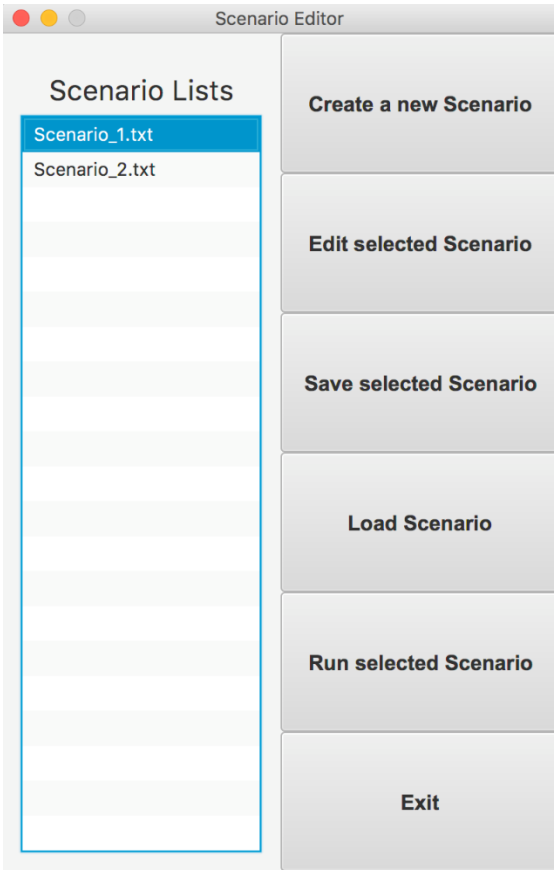

Fig 6. Scenario Editor after Loading Scenarios

Then, with Scenario_1.txt highlighted, click on 'Run selected Scenario'. Alternatively, you may run Scenario_2.txt. After clicking on 'Run selected Scenario', you will be prompted to choose between a visual player or an audio player as shown in figure 7.

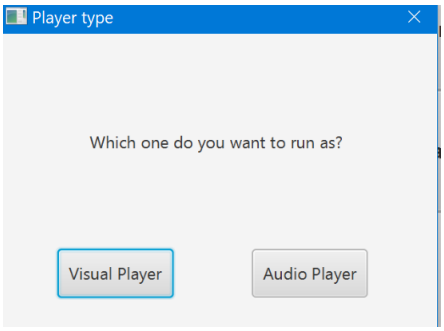

Figure 7. Visual or Audio prompt

For now, choose the visual player. A simulated braille cell should appear with buttons labeled 1,2,3 and 4 at the bottom for user interaction. The following figure illustrates this (figure. 8):

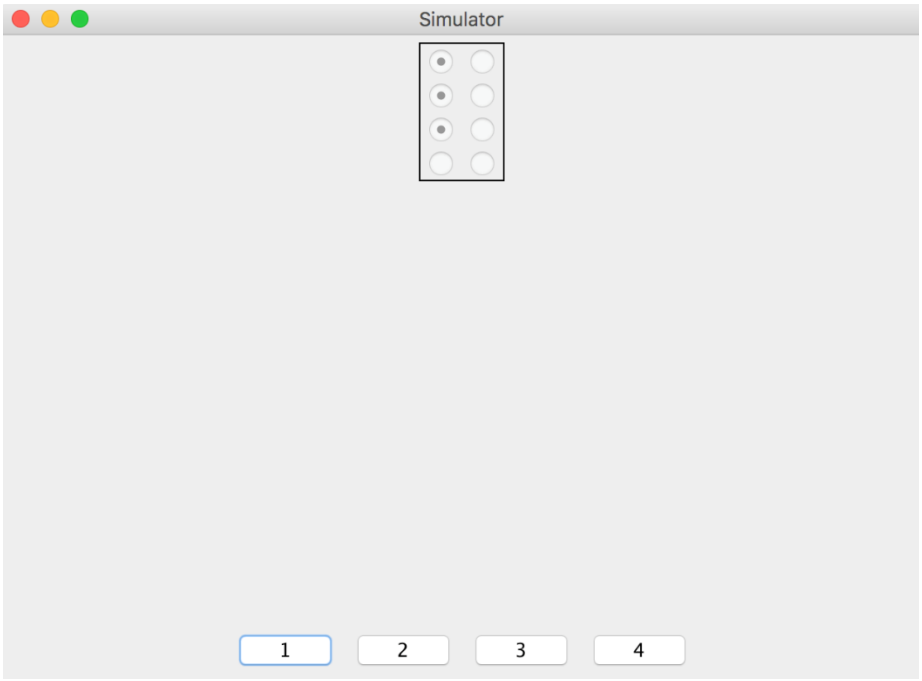

Fig.8 Simulated braille cells

At this point you may try this scenario and play around with the input to familiarize yourself with how a basic run of the application may proceed.

For instance, let us quickly run through Scenario_2. First, note that pins 1-3 are the first 3 cells on the left and pins 4-6 are the first three on the right.

After running Scenario_2, you will be asked if pins 1,3 versus pins 4.6 are highlighted (figure.9).

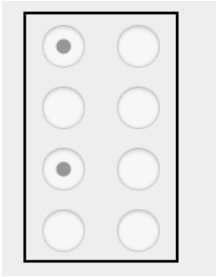

Fig.9 Pin alignment for 1,3 versus 4,6

These are pins 1,3. Therefore, click on button 1. A bell sound should be heard. The bell signifies correct answers.

Next, you will be asked about this configuration (figure.10):

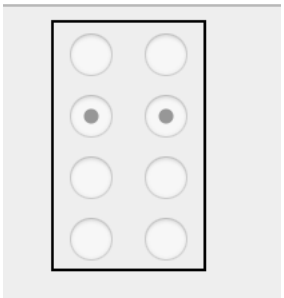

Fig.10 Pins 2,5 versus 1,4

These are pins 2 and 5. However, let us choose the incorrect option of pins 1 and 4 by clicking button labeled ‘2’. You should hear an unappealing buzzer sound. This buzzer sound signals an incorrect answer. The rest of the scenario runs similarly. You may finish it if you desire.

The following section will introduce the capability of creating your own scenario.

5.0 Scenario Editor: Creating and Editing a New Scenario

Creating a new Scenario

After exiting the simulated braille cells from section 4.0 by clicking the red circle or the analogous button on your system, you will be able to use features of the Scenario Editor from figure 4 on page 8.

The Authoring Application will allow the user to create scenarios such as those seen in section 4.0. Click on 'Create a new scenario'. The following display should appear (figure 11):

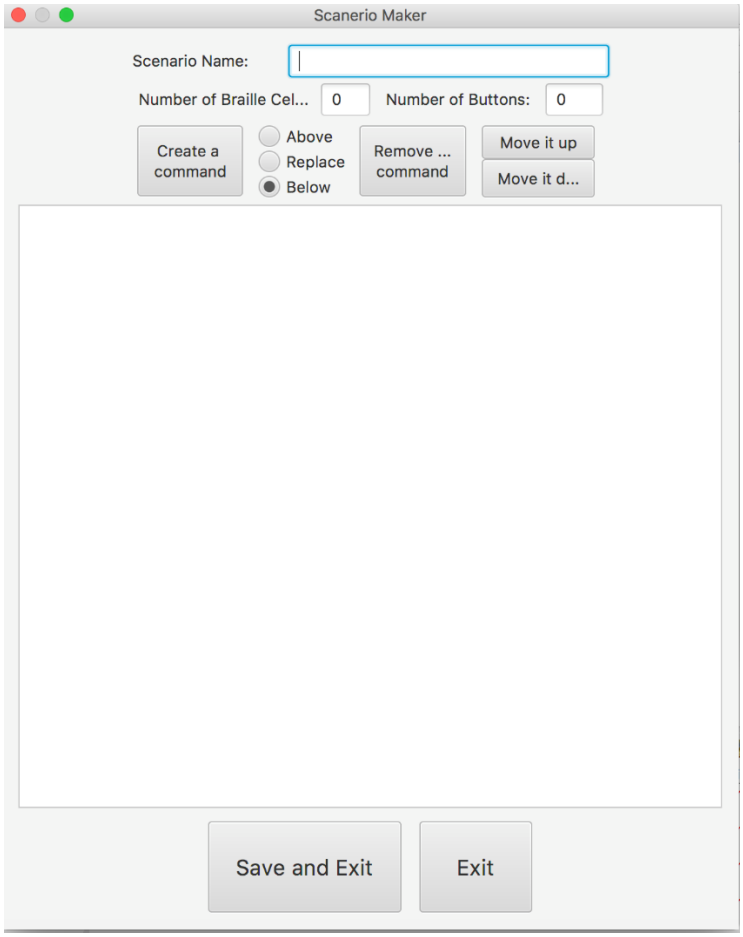

Fig.11 Scenario Maker

This portion of the application will allow the user to create their scenario in the text box located in the center of the display. The user will begin by clicking on 'Create a Command' located in the bottom row of items to the far left near the top of the display. A drop box will appear where you can select the type of command to produce. Figure 12 depicts an added display string.

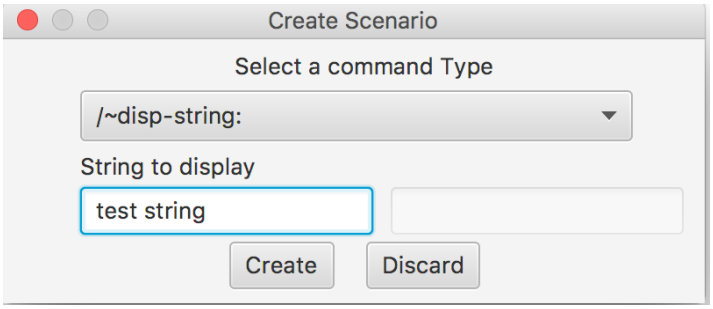

Fig. 12 Adding a display string command to new scenario

Explore the various commands available. As a precondition, it is assumed the user understands how to create a syntactically valid scenario.

Multiple commands can be added and manipulated. For instance, figure 13 shows a skip command added after the previous display command. Then, figure 14 utilizes the 'Move it down' command to interchange the skip and display command. Moreover, you can specify a braille cell configuration in the 'Number of Braille Cells' and 'Number of Buttons' boxes.

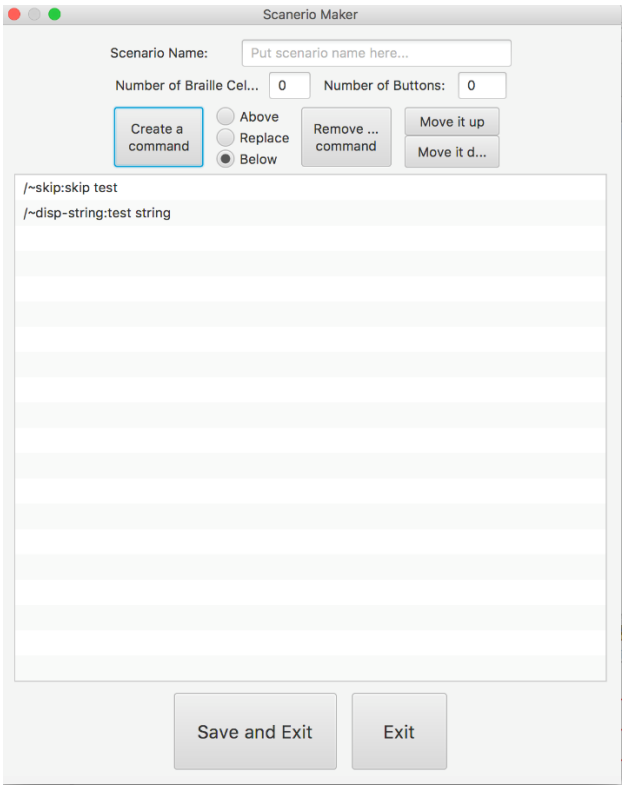

Fig.13 Adding a skip command

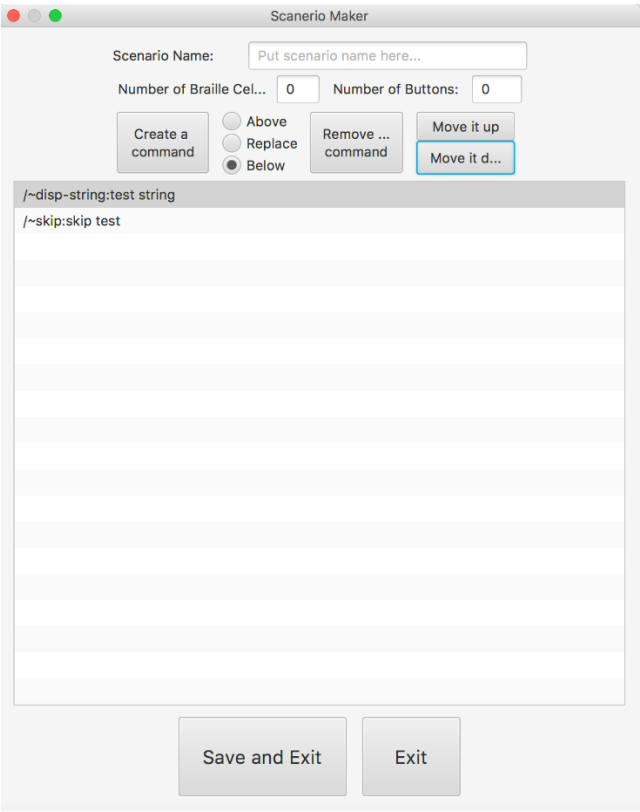

Fig.14 Moving the skip command downwards

Additionally, you can remove commands by clicking on the 'Remove command' button. When finished, enter a name in the 'Scenario Name' textbox and click the 'Save and Exit' button. Your scenario can now be loaded and executed. Please refer to section 4.0 on loading scenarios.

Editing a new Scenario

Let us return to the scenario editor as depicted in figure 4 on page 8. You will notice a button labeled ‘Edit Scenario’. This mode enables the user of the application to launch an existing scenario’s contents in an editor pane. Simply load a file and highlight, or highlight an existing file already appearing in the display and click on ‘Edit Scenario’. The interface is like the ‘Create a New Scenario’ panel. Figure 15 highlights the editor pane.

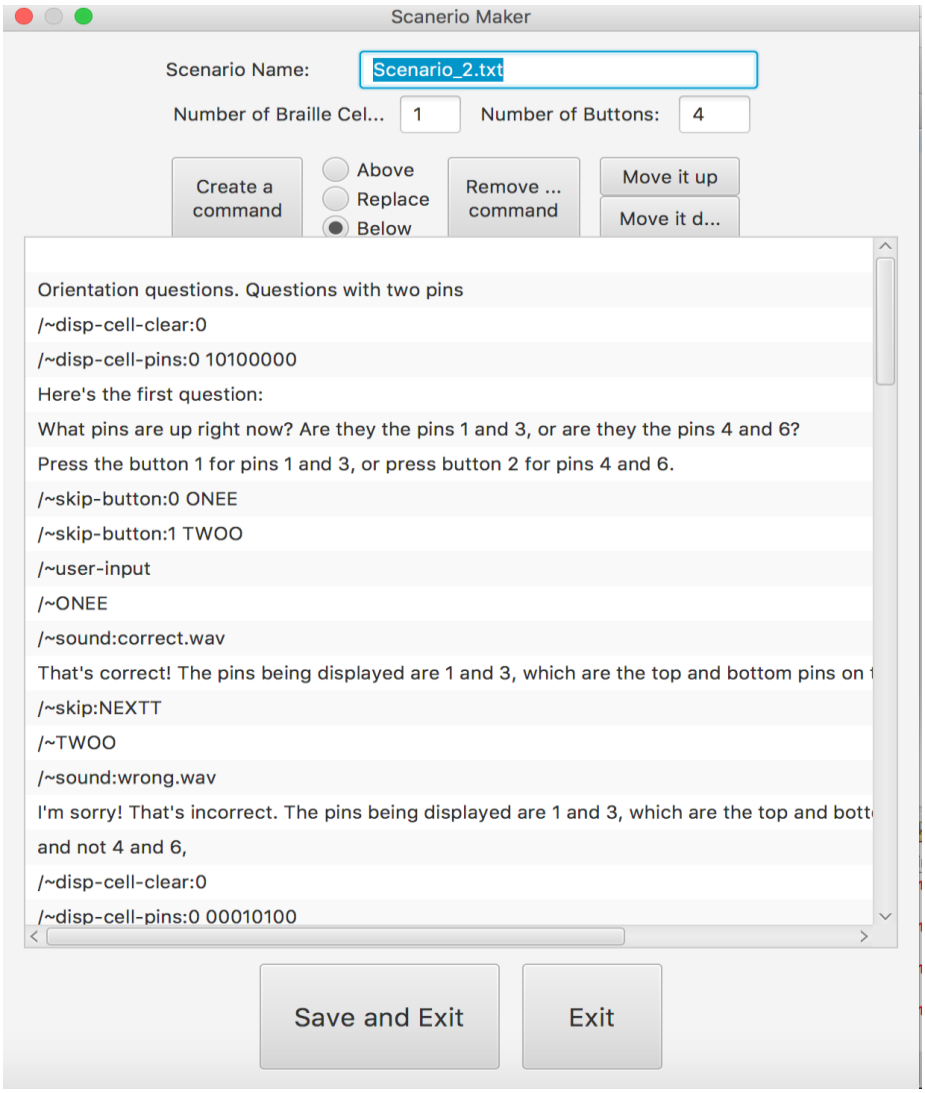

Figure 15. Edit Scenario

Now you will be able to alter the file by creating new commands, re-arranging commands or deleting commands. When you are complete, click on ‘Save and Exit’.

The remaining modes in the scenario interface ‘Save Selected Scenario’ and ‘Remove Selected Scenario’ act as titled.

6.0 Voice Recorder

In the event the scenario editor or visual player are not feasible for use by a user who is visually impaired, the Voice Recorder mode can be used to communicate directives.

At the launch of the application seen in figure 2 on page 7, choose 'Voice Recorder'. The user interface illustrated in figure 16 will appear.

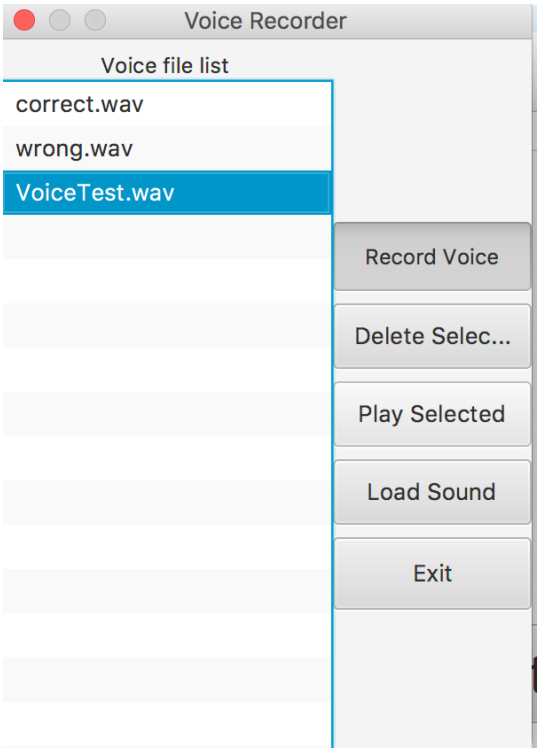

Fig. 16 Voice Recorder

The files correct.wav and wrong.wav will appear by default. VoiceTest.wav is a user created sound. You can listen to the default files by highlighting one and clicking on 'Play Selected'. You will recognize these sounds from section 4.0.

In this example, we utilized the 'Record Voice' button which prompts for a file name for the audio file being created. 'VoiceTest.wav' represents the audio file. Figure 17 depicts the instantiation of this file. Thereafter, the 'Load Sound' button was used to add VoiceTest.wav to the voice file list. Figure 18 exposes the window seen when clicking on 'Load Sound'. Again, use the 'Play Selected' button to play the audio file you just recorded.

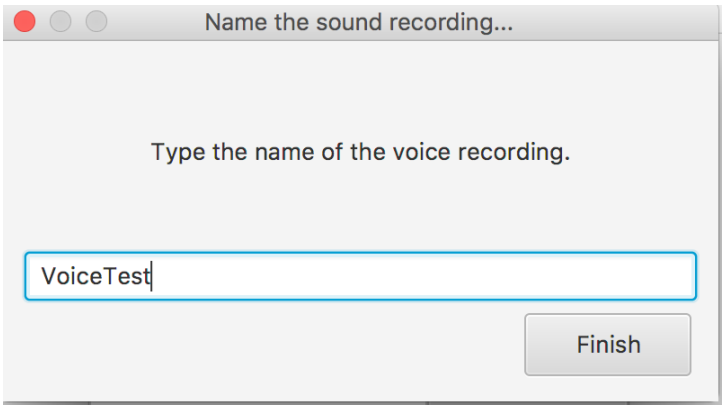

Figure 17. Creating VoiceTest.wav

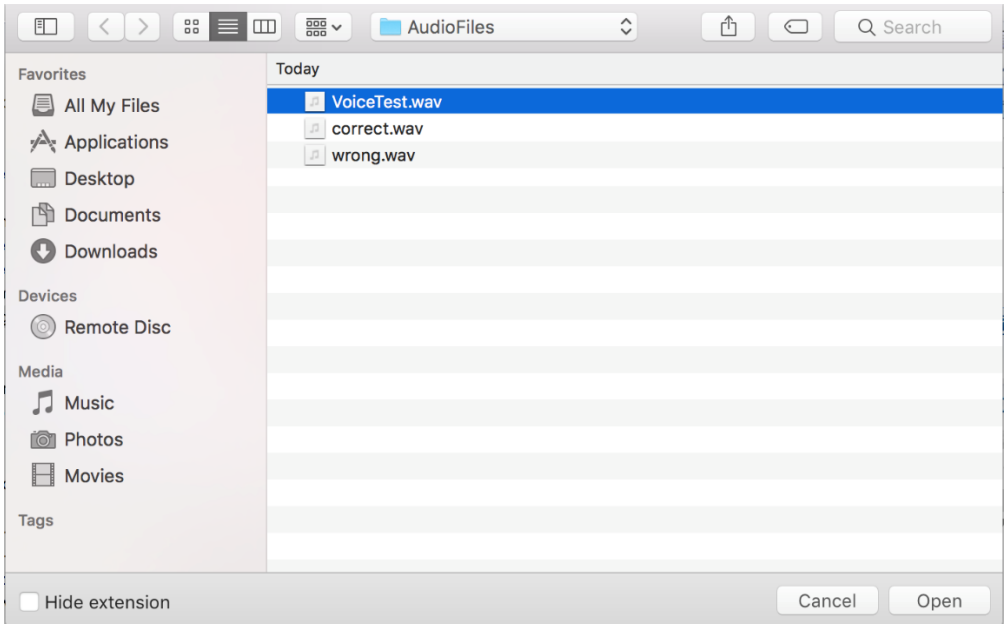

Figure 18. Load Sound window

Finally, the 'Delete Selected' button can be used to remove voice files no longer needed.

7.0 Screen Reader Compatibility

Note that the manual will be revised with a full specification for this section. This functionality is not fully implemented as the application is currently in a beta stage. However, the application will enable the operating system's built in screen reader. Specifically, this application will support Windows NVDA and the Mac's Voiceover.

The following video we created illustrates how text can be read by the VoiceOver on the iOS operating system. This demo uses a JFileChooser from the Swing library and will not represent implementation of the finished application. Please follow the link below or copy and paste it into a browser. Connection to the internet is required:

http://www.youtube.com/watch?v=RcSPowKO8Yc

Video 1. VoiceOver iOS