Authoring Application

Testing Document

Group 2

EECS 2311

Final Submission

August 21st 2018

1. Table of Contents

2. Purpose

The purpose of this document is to detail the testing specifications of our Authoring application.

2.1. Definitions

In this document, the following terms are defined as follows:

Scenario: *an interactive activity or lesson to be performed with the braille cell.*
Interaction: *any of the features outlined in the scenario file format document.*
User: *a user of the authoring application software. E.g. a braille instructor*
Student: *a user of the braille cell; i.e. a student learning braille through the braille cell, one whom is the intended user of the lessons created through this app.*
Braille Interface: *the braille cell simulator*
Authoring Application (AA): *the application where instructors can create or edit scenarios*
Entry Point (EP): *the initial window that pops up upon executing the program*
Interaction List (IL): *a list containing the various interactions in a given scenario*
Configuration Panel (CP): *a pane in the AA that allows users to configure the interaction based on their input. E.g. specifying text to be read to a*
Blind*: *those who are blind or with low vision*

3. Test Requirements

The following testing requirements are derived from the requirements document and will serve as the baseline for determining our test cases in section 3 below.

3.1. Scenario Requirements

3.1.1. Open Scenario File

The user shall have the capability to open an existing, properly formatted, scenario text file into the application.

3.1.2. New Scenario File

The user shall have the capability to create a new scenario file, specifying the number of cells, buttons, and title of the scenario.

3.1.3. Edit Scenario File

Upon opening or creating a new scenario file, the user shall have the capability to reorder, add, and remove interactions from the scenario. The user shall also be able to select interactions in the scenario and edit the contents of the selected interaction.

3.1.4. Run Scenario File

The user shall be able to run their scenario in the simulator in order to test their scenario out.

3.1.5. Save Scenario File

The user shall be able to save the scenario they have created or edited within the application to a valid scenario text file.

3.2. Interaction Requirements

3.2.1. Read

The read interaction shall be an interaction where the simulator reads out the user's specified text to the student. The user shall be able to edit the text read out to the student. This will be verified by creating a new read interaction and writing sample text into it, and finally ran and correctly read out by the simulator.

3.2.2. Voice

The voice interaction shall be an interaction where the simulator plays a recorded message to the student. The user shall be able to record from within the application and delete/re-record if they choose to do so. This will be verified by creating a new voice interaction and recording a small sample recording, then checked by testing it in the simulator.

3.2.3. Pause

The pause interaction shall be an interaction where the simulator pauses for a user-defined number of seconds. This will be verified by timing the length of the pause interaction in a sample scenario and ensuring it aligns with the specified number of seconds in the interaction.

3.2.4. Display Braille

The display braille interaction shall be an interaction where the simulator displays a user-defined Braille symbol on a cell of the user's choosing. This will be verified by seeing whether the simulator displays the correct braille symbol based on the user generated interaction.

3.2.5. Keyword

The keyword interactions links the application between different parts of the scenario file, linked by a user-specified keyword. This will be verified by attempting a skip/keyword interaction in order to skip over a certain read interaction. If the read interaction was not read, then it will be determined successful.

3.2.6. Skip Button

The skip button interaction is used to link a button press with a certain keyword interaction This interaction is used in combination with the user-input interaction. This will be verified by creating a test scenario where only a specific read interaction is read when button 1 is pressed, and only a specific read interaction is read when button 2 is pressed.

3.2.7. Skip

The skip interaction is the second half of the skip/keyword interaction, where users can use this to skip to a different section of an interaction. See Keyword's test case for this interactions acceptance case.

3.2.8. User Input

The user input interaction is how users can pause the program and wait for a student to push a button in the simulator. The program will skip to the linked skip-button interaction. This will be tested as outlined in the skip-button section.

3.2.9. Reset Button

The reset button interaction is used to clear the listeners for all the buttons in the simulator. This will be tested by verifying that the simulator doesn't respond to further button presses after a skip-button/user-input combination is used with a reset-button afterwards.

3.2.10. Cell Clear

The cell clear interaction is used to clear the display of all the braille cells. This will be verified by inserting a reset button interaction into a scenario where display braille was called before it, and viewing the braille-cells clearing their raised pins. [1]

4. Test Cases

ID	Description	Procedure	Expected Output	Result
Testing main panel				
TC1	Validating requirement 3.1.1 - Open Scenario File	1. Click **Open** button from main page 2. Choose a scenario file from the file dialog 3. Click **Open**	If a valid scenario file was selected, the editor screen should appear with all the interactions listed and the configurations of the selected interaction. If an invalid scenario file was selected, an error message dialog will notify that an invalid scenario file was chosen. User will remain on the startup screen on dialog close.	Pass
TC2	Validating requirement 3.1.2 - New Scenario File	1. Click **New** button from main page 2. On the New scenario page, enter the scenario title,	On clicking Create, the editor page should appear with an empty interaction list.	Pass

		and set the number of cells and buttons as desired. 3. Click **Create**		
Testing editor panel				
TC3	Validating requirement 3.1.3 - Edit scenario file	Create / edit: 1. Create a new scenario from the main page 2. On the editor page, select an interaction from the drop-down box (on the bottom with the other controls) 3. Click **Add** Load / edit: 1. Open a scenario file from the main page 2. Select any interaction from the list 3. Click any of the controls / fields associated with the interaction on the configuration panel on the right to change them	The configuration options for the selected interaction should appear on the right. The fields associated with the particular interaction are editable.	Pass
TC4	Validating requirement 3.1.4 - Run scenario file	1. Create a new scenario or load an existing one from the main page 2. If a new scenario was created, add any interaction to test and click **Save**	The simulator window should appear with the computerized voice playing audio.	Pass

		3. Click **Run** to launch the simulator		
TC5	Validating requirement 3.1.5 - Save scenario file	1. Create a new scenario or load an existing one from the main page 2. Make changes or add new interactions to the scenario 3. Click the **Save** button	If a new scenario was created, a new .txt file with the title of the scenario will be saved under a folder called **Scenarios** in the working directory of the application. If a scenario was loaded, it will overwrite the existing .txt file.	Pass
Testing interaction configurations and behavior				
TC6	Validating interaction requirement 3.2.1 - Read	1. Create a new scenario from the main page 2. Select **Read** from the drop-down list of interactions 3. Click **Add** 4. On the configuration pane on the right, modify the **Data** field on the right to the text that simulator should read out 5. Click **Save** 6. Click **Run**	The simulator should open and read out the text data that was specified for the **Read** interaction.	Pass
TC7	Validating interaction requirement 3.2.2 - Voice	7. Create a new scenario from the main page 8. Select **Voice** from the drop-down list of interactions 9. Click **Add** 10. On the configuration pane on the right,	The simulator should open and play back the audio clip that was attached with the **Voice** interaction. The audio (.wav) file should be saved in its appropriate location.	Pass

		click the **Record** button to begin recording an audio with the active microphone. Click **Stop** to stop recording. Or click **Browse** to load an audio file (.wav) and skip to step 12. 11. Save the audio file to a location on your local drive using the Save dialog 12. You can play back the audio using the controls on the editor pane (play, pause). 13. Click **Save** 14. Click **Run**		
TC8	Validating interaction requirement 3.2.3 - Pause	1. Create a new scenario from the main page 2. Add a **Read** interaction and set the desired text to be read out 3. Select **Pause** from the drop-down list of interactions and click **Add** 4. Select the number of seconds to pause for 5. Add another **Read** interaction and modify the data field 6. Click **Save** 7. Click **Run**	The simulator should open and read out the first set of text that was set, and then wait for the selected amount of seconds, and then finally read the second set of text.	Pass

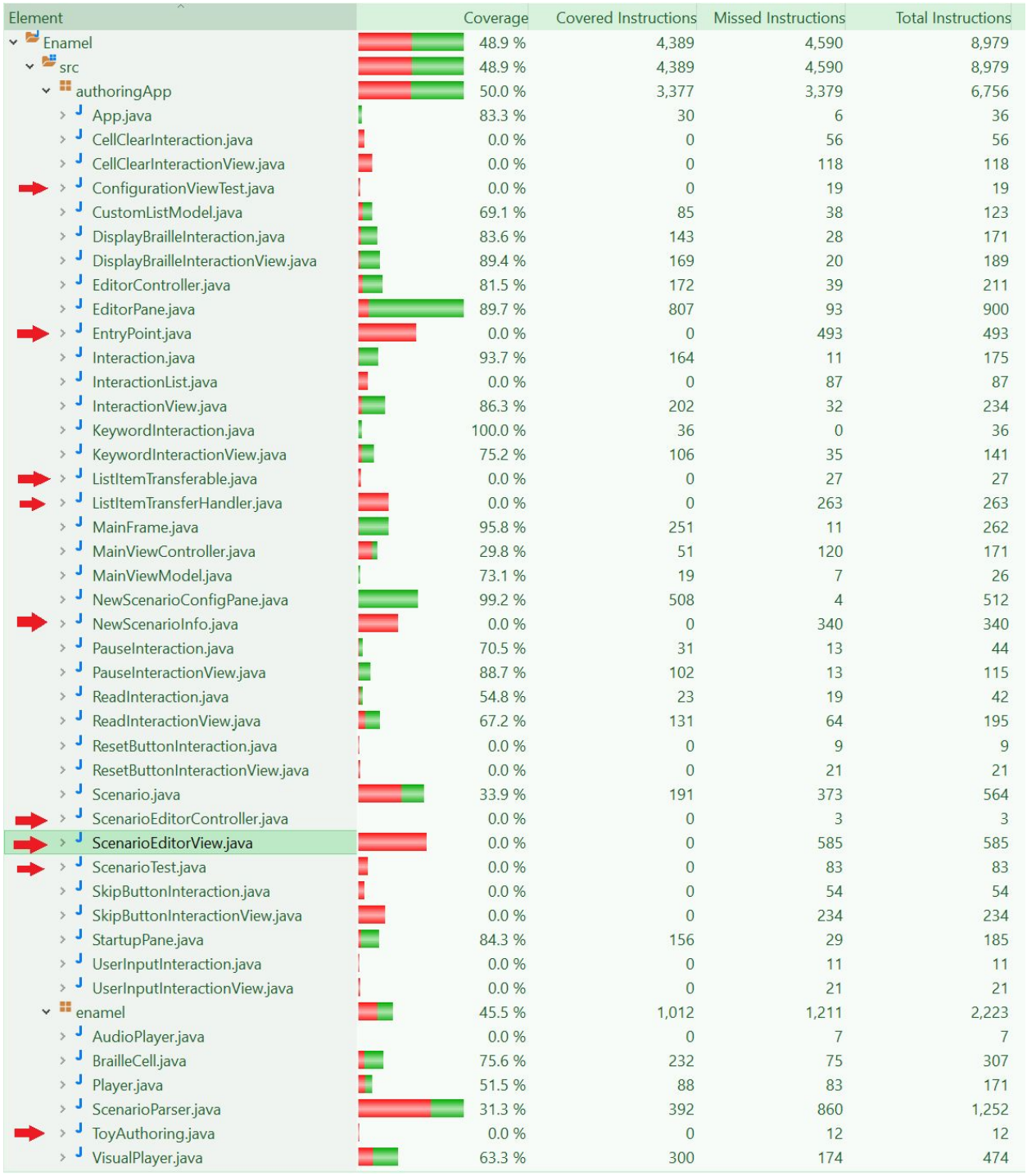
TC9	Validating interaction requirement 3.2.4 - Display braille	1. Create a new scenario from the main page 2. Select **Display braille** from the drop-down box of interactions and click **Add** 3. Select / deselect the pins (combo-box items) to raise on the simulator 4. Click **Save** 5. Click **Run**	The simulator should open with the selected pins being shown on the braille pad.	Pass
TC10	Validating interaction requirement 3.2.5 - Keyword	1. Create a new scenario from the main page 2. Select **Keyword** from the drop-down box of interactions and click **Add** 3. Set a keyword to skip to a line in the scenario where the keyword is used 4. Click **Save** 5. Continue to TC11 to complete testing this interaction.	The Keyword interaction, by itself will not do anything. No output expected except the addition of the interaction to the interaction list.	Pass
TC11	Validating interaction requirement 3.2.6 - Skip Button	1. Add several random interactions (**Read**'s are nice and simple) 2. Select **Skip Button** from the drop-down box of interactions and click **Add** 3. Set a button number that the	The simulator should open, read off the couple **Read**'s (if they were the interactions used for this example), then if the user presses the set button number, it will skip the following **Read**'s to the step with the **Keyword** and begin performing the interactions following (in this case the final read)	Pass

		user should press and the previously added **Keyword** to jump to. 4. Reorder the interactions to have a few reads, then the **Skip Button**, then a few **Read**'s, the **Keyword**, then finally a final **Read** to show we've reached this part. 5. Click **Save** 6. Click **Run**		
TC12	Validating interaction requirement 3.2.7 - Skip	1. Create a new scenario from the main page 2. Select **Skip** from the drop-down box of interactions and click **Add** 3. Set a keyword to skip to a line in the scenario where the keyword is used 4. Perform the steps outlined in TC11 (adding random interactions) to observe the **Skip** interaction 5. Click **Save** 6. Click **Run**	The simulator should open, perform the interactions prior to the **Skip** and then skip to the part where the keyword is used and continue performing interactions following it.	Pass
TC13	Validating interaction requirement 3.2.8 - User Input	1. Create a new scenario from the main page 2. Select **User Input** from the drop-down box of interactions and click **Add**	The simulator should open and pause, waiting for a button press. Once the button is pressed, it will continue performing the remaining interactions.	Pass

		3. Add a few interactions below (**Read** for simplicity) 4. Click **Save** 5. Click **Run**		
TC14	Validating interaction requirement 3.2.9 - Reset Button	1. Open an existing scenario with the **Skip Button** (from TC11) 2. Add another **Skip Button** towards the end of the interaction list and a few **Read**'s following. 3. Select **Reset Button** from the drop-down box of interactions and click **Add** 4. Reorder the Reset Button interaction to be right above the 2nd **Skip Button** 5. Click **Save** 6. Click **Run**	The simulator should open, read off the couple **Read**'s (if they were the interactions used for this example), then if the user presses the set button number, it will skip the following **Read**'s to the step with the **Keyword** and begin performing the interactions following. Before it has reached the 2nd Skip Button, it will clear attachments of the buttons and keywords. Once it reaches the 2nd **Skip Button**, pressing the button will not skip to the previously set keyword and instead will continue to the end.	Pass
TC15	Validating interaction requirement 3.2.10 - Cell Clear	1. Open an existing scenario file with the **Display Braille** interaction (TC9) 2. Select **Clear braille** from the drop-down box of interactions and click **Add** 3. Set the cell number to clear. 4. Click **Save** 5. Click **Run**	The simulator should open, display the braille, and then clear the cell number specified of any set pins.	Pass

5. Test Coverage

The metrics available at this time for our testing include the test coverage results from running the authoring application and performing each function available in the application.



Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
Enamel	48.9 %	4,389	4,590	8,979
src	48.9 %	4,389	4,590	8,979
authoringApp	50.0 %	3,377	3,379	6,756
App.java	83.3 %	30	6	36
CellClearInteraction.java	0.0 %	0	56	56
CellClearInteractionView.java	0.0 %	0	118	118
ConfigurationViewTest.java	0.0 %	0	19	19
CustomListModel.java	69.1 %	85	38	123
DisplayBrailleInteraction.java	83.6 %	143	28	171
DisplayBrailleInteractionView.java	89.4 %	169	20	189
EditorController.java	81.5 %	172	39	211
EditorPane.java	89.7 %	807	93	900
EntryPoint.java	0.0 %	0	493	493
Interaction.java	93.7 %	164	11	175
InteractionList.java	0.0 %	0	87	87
InteractionView.java	86.3 %	202	32	234
KeywordInteraction.java	100.0 %	36	0	36
KeywordInteractionView.java	75.2 %	106	35	141
ListItemTransferable.java	0.0 %	0	27	27
ListItemTransferHandler.java	0.0 %	0	263	263
MainFrame.java	95.8 %	251	11	262
MainViewController.java	29.8 %	51	120	171
MainViewModel.java	73.1 %	19	7	26
NewScenarioConfigPane.java	99.2 %	508	4	512
NewScenarioInfo.java	0.0 %	0	340	340
PauseInteraction.java	70.5 %	31	13	44
PauseInteractionView.java	88.7 %	102	13	115
ReadInteraction.java	54.8 %	23	19	42
ReadInteractionView.java	67.2 %	131	64	195
ResetButtonInteraction.java	0.0 %	0	9	9
ResetButtonInteractionView.java	0.0 %	0	21	21
Scenario.java	33.9 %	191	373	564
ScenarioEditorController.java	0.0 %	0	3	3
ScenarioEditorView.java	0.0 %	0	585	585
ScenarioTest.java	0.0 %	0	83	83
SkipButtonInteraction.java	0.0 %	0	54	54
SkipButtonInteractionView.java	0.0 %	0	234	234
StartupPane.java	84.3 %	156	29	185
UserInputInteraction.java	0.0 %	0	11	11
UserInputInteractionView.java	0.0 %	0	21	21
enamel	45.5 %	1,012	1,211	2,223
AudioPlayer.java	0.0 %	0	7	7
BrailleCell.java	75.6 %	232	75	307
Player.java	51.5 %	88	83	171
ScenarioParser.java	31.3 %	392	860	1,252
ToyAuthoring.java	0.0 %	0	12	12
VisualPlayer.java	63.3 %	300	174	474

Figure 1. Screenshot of test coverage results

6. References

1. C. Dear, *Authoring Application Requirements Document*, 1st ed. 2018.