Testing Documentation

Authoring App for Treasure Box Braille

Table of Contents

Testing

Introduction:

To ensure accuracy and satisfactory performance of the application, we have outlined what functions need to be tested. It is especially important to test that every braille prompt is correct in its representation. We also test to ensure the application follows the correct path, particularly in the case of branching (i.e., non-linear layout).

In our preliminary launch of the application, we do not expect to have all testing in place. Much of the application lies in the graphical user interface, which in our determination, is best tested visually by running the application.

Implemented Tests:

Test	Expected	Actual
Braille character representation	Each letter of the English alphabet is correctly represented in Braille	Only lower-case letters of the English alphabet so far correctly depicted
Graphical structure integrity	Each time a button or node is added, it is visually and audibly updated by the application	We are unable to test this at the moment
Button prompts are followed	Scenario is followed correctly as directed by user	We are unable to test this at the moment
Screen reader reliability	Every graphical element is relayed audibly to the user	Graphical elements have accessibility contexts read aloud

Rationale and Sufficiency:

It would not be difficult for non-braille users (in this case, the developers) to overlook any incorrect braille representations. As such, it would be particularly important to test and ensure the accuracy of any braille represented in our application.

The application also incorporates a small number of menu prompts, leading to different "areas" of the program. For example, on the main menu screen upon loading of the application, the user has the option to create a new scenario file. We would ensure that upon selecting that option (or any option for that matter), the application updates the text and buttons on the screen to be consistent with the part of the program that facilitates the creating of a new scenario.

As such, with the above tested for and the scenario being adhered to as expected, the basic functionality of the application would be achieved.

Furthermore, we test for correct audio cues being played when a particular prompt is selected, such as a "correct ding" audio file being played when the correct response to a question is selected.

Lastly, accessibility context is provided for all visual graphical elements that are present on the screen. As such, anytime a graphical component (such as a node, or the text within the node) is updated, added, or removed, this information is relayed audibly to the user. We wish to ensure that every graphical element relays this audible element.

Test Coverage:

Due to some parts of the application still being under construction, we are unable to reliably reach a high percentage of coverage. Some classes are unable to be run, and thus tested for. For those methods that can be run, we mostly rely on running the program manually rather than JUnit testing as much of it is graphical in nature.

Future and Improvement:

We are fully aware of the incomplete state of the application, which heavily impacts our ability to create a test suite that checks all aspects as desired. As such, we rely on manually checking the reliability of our application as opposed to the more ubiquitous method of using a testing suite.