Testing Document

Connor Ahearn, Micah Arndt and Kevin Chan

Format

The format of the document is structured as follows:

- The class / function being tested
- Summary of the class / function's purpose or task
- Test cases implemented for said class / function
    - Also listing why these cases were created
- Why the cases listed above are sufficient

Printer.java

The printer class is the portion of the application tasked with printing the information from the GUI onto a text file. It does this by receiving information in the form of *Block* objects (**Block.java** listed above). It then stores the information in a collection until the **print()** method is called, which is when the information is printed to the text file. The text file it produces is based on the examples given for the simulator to run.

The class itself doesn't serve a complex purpose. It has one specific task, and as such only has a public constructor and 2 public methods.

Printer() (Constructor)

- Creates a new Printer object
    - Takes a filename for the new file, as well as how many cells and buttons the new scenario will use
    - Places the information required for the first 3 lines of the text file into the **lines** collection

The constructor was tested with the **testInitial1()** test in **testPrinter.java**. It printed the input to the text file in the correct format. The same test was used throughout the rest of the tests to assure the initial block is correct before any other information is printed.

addBlock() and *print()* (Method)

- Adds a block object to the collection of lines to be printed
- Formats the information in a way the BrailleBox can read it
- Prints the information in the collection to the text file

These methods were tested in the **test1Block()** and **test2Block()** methods. The main portion being checked for the addBlock() method is that the information is readable for the BrailleBox, and no information is lost. These tests checked the correctness of the text file printed for 1 and 2 blocks of input.

More tests will be required for these two methods in the future when more types of information are accepted, but for now the current test structure is adequate.

Block.java

Object based on storing data for a question / story in a Scenario. Just has a constructor and public fields for the time being. Currently no functionality for changing information in a block in the application. The constructor initializes all fields via parameter values. Any tests written for this class at this point would be redundant, but when more functionality is added this will change.

BrailleInterpreter.java

Object that converts characters to braille pin binary equivalent. Braille pin equivalents are stored and retrieved from a HashMap, and the pin equivalents were based on the BrailleBox simulator code. Any tests on this class would be testing either the Java HashMap implementation itself or the tested code provided to us by the professor. Either way, these tests would be redundant.

AudioPlayer.java & ScenarioCreator.java

These classes are both not fully implemented and cannot be tested via JUnit. Manual tests will be written out for these classes when their implementations are complete.