Testing Document

Test Cases

The first set of test cases check the function of the save and load method as well as the buttonCatalog. A test case was made to ensure that the serialized file is loaded correctly. It checks by ensuring the loaded object has the proper state. Assertions will check actual button data. Another test case was created to ensure that if the file did not exist, the proper exception was thrown. These tests are sufficient as it only involves two methods, save and load, and the main object of interest is the button.

The next test will assure that the correct buttons are loaded. This will be done by creating a mock button in the gridPane. Upon loading of the button, an output will be created and that will be tested with an assertion. This is sufficient to test the functionality of the GUI, because it is one of the only dynamically generated layouts that all other functions depend on.

Most of the methods in the provided interface will be tested as well. Most of the methods are getters for state, and will be checked to ensure the correct state of the object. This will ensure functionality of the implemented interface and is sufficient for this case.

Another series of text cases will ensure that audio files are created upon compile. It will test the count and name, compared to the number of buttons in the catalog. This is sufficient to ensure proper GUI layout loading since as long as there are the right number of buttons, and right number of states, the GUI has loaded properly.

The functionality of the talkbox was purposely constrained, as this is only the minimum viable product. The only user input, is button clicks and textfield input. Those are the main sources of variability. There is not a lot of room for errors to occur, unless the underlying code and implementations are incorrect. Even in this case, the pre and post conditions maintain valid state within the program.