Test Cases

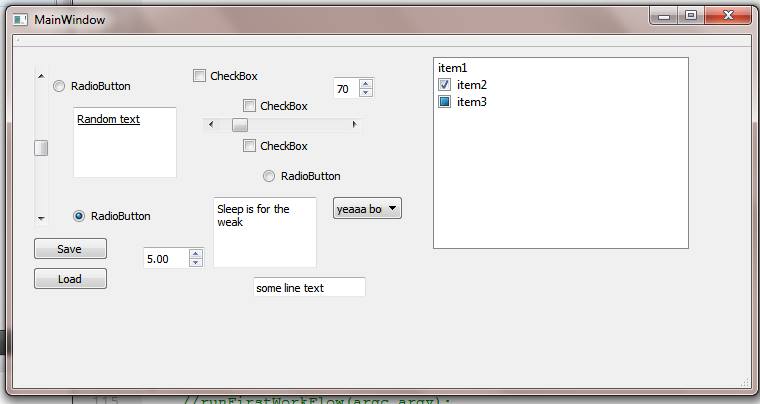
We will test the software based on how well it captures the parameters, how it stores them in a generated XML file, how it loads that same XML file, and how the same GUI can have that loaded in to it during another session in the Qt Creator. We want to make sure that none of the data is lost during the save process and that it can be loaded without having any issues in reading the XML.

Our client, Francois Budin, will assess the final testing of our project. If any issues occur, we can correct them and give the updated version to him to continue testing. As of right now, we plan on giving him a week before the final product is due. We plan to ask Francois to see if this is enough time for him to make sure that the product works fine on a daily basis.

The following Qt GUI widgets have been tested and marked as fully functional (they are stored in the model as their respective data types):

* QTextEdit
* QCheckBox
* QRadioButton
* QComboBox
* QSpinBox
* QDoubleSpinBox
* QScrollBar
* QLineEdit

The following is the GUI that we designed and tested to ensure that all of the widgets listed above are saved and loaded as expected.



**Ideal Plan**

Platforms to be tested

* Qt Creator

More widgets to be added

* QListWidgets
* QTabWidgets

Test Cases

* Does the program only parse UI files and what other data file types can it be applied to?

                        -It can be applied to any well-formed XML document.

* Is there anyway that the program can run without having all of its generated files in the same directory as the user's Qt project?

-It is possible, but for convenience purposes all of our generated files appear in the same directory.

* Is there anyway that the software can run independently without having to be used through the Qt Creator?

                        - It is possible using Cmake, but Qt Creator makes it easier.

* What other Qt widgets can this software support (all of them would be ideal)? For any of the widgets that may not be supported, is there an alternative way to capture their parameters?

- Some of the widgets are arbitrarily expandable and each expanded item has corresponding properties so all of the items along with their properties would have to be packaged into a data structure corresponding to the entire widget. Particular examples of such widgets (not supported) are QListWidgets and QTabWidgets, both of which exhibit an extendable and nested attribute structure.

**Intended Plan**

Platforms to be tested

* Qt Creator

Test Cases

* Can our software handle nested elements?

- Widgets with nested elements are not handled. Such examples are QListWidgets and QTabWidgets.

* If an XML representation is created by the saver, an element is removed from the GUI, a new model is created, can the data be reloaded except the data for recently removed element?

- Yes, when the first workflow is run, it will detect the change in the GUI XML and adjust its data representation accordingly so that when it is reloaded in the second workflow, it will be loaded properly.

* Can either the saver, loader, or the model be regenerated if one of them were to be deleted?

                - Yes, the first workflow regenerates all three files from the GUI XML.

* Can the software support developer manually make model, saver and loader files?

- Yes, as a matter of fact, that is what our client currently does. As GUIs become more complicated and more widgets are added, however, this approach no longer remains feasible. As a result, the automated generation of the model, saver, and loader files that our program enables saves the client a lot of work. It is in fact a usable piece of software for anyone using Qt Widgets who wants to be able to save a data representation and so it has potential to become useful open-source code.