Using a Designer UI File in Your Qt for Python Application

Converting the Form to Python Code

To demonstrate, we use the Qt Widgets animation easing example.

The application consists of one source file, `easing.py`, a UI file `form.ui`, a resource file `easing.qrc` and the project file, `easing.pyproject` file in the YAML format:

```
{
    "files": ["easing.qrc", "ui_form.py", "easing.py", "easing_rc.py",
              "form.ui"]
}
```

The UI file is converted to Python code building the form using the User Interface Compiler (uic):

```
uic -g python form.ui > ui_form.py
```

Since the top level widget is named `Form`, this results in a Python class named `Ui_Form` being generated. It provides a function `setupUi()`, taking the widget as parameter, which is called to create the UI elements:

```
from ui_form import Ui_Form
...
class Window(QtWidgets.QWidget):
    def __init__(self, parent=None):
        super(Window, self).__init__(parent)

        self.m_ui = Ui_Form()
        self.m_ui.setupUi(self)
```

Later on, the widgets can be accessed via the `Ui_Form` class:

Besides `setupUi()`, `Ui_Form` provides another method `retranslateUi()`, which can be called in reaction to a QEvent of type QEvent.LanguageChange, which indicates a change in the application language.

The UiTools Approach

The QUiLoader class provides a form loader object to construct the user interface at runtime. This user interface can be retrieved from any QIODevice, e.g., a QFile object. The QUiLoader::load() function constructs the form widget using the user interface description contained in the file.

It is demonstrated by the uiloader example:

```
from PySide2.QtUiTools import QUiLoader

if __name__ == '__main__':
    # Some code to obtain the form file name, ui_file_name
    app = QApplication(sys.argv)
    ui_file = QFile(ui_file_name)
    if not ui_file.open(QIODevice.ReadOnly):
        print("Cannot open {}: {}".format(ui_file_name, ui_file.errorString()))
        sys.exit(-1)
    loader = QUiLoader()
    widget = loader.load(ui_file, None)
    ui_file.close()
    if not widget:
        print(loader.errorString())
        sys.exit(-1)
    widget.show()
    sys.exit(app.exec_())
```

‹ Using a Designer UI File in Your C++ Application

Using Custom Widgets with Qt Designer ›