QtWidgets and QtQuick.Controls - A Comparison

Qt Developer Days Europe 2014

Presented by Kevin Krammer
kevin.krammer@kdab.com
• The Question

• Side-by-Side Comparison

• Conclusions
What should I use for a new project: 
**QtWidgets** or **QtQuick.Controls**?
It depends!
• The Question

• **Side-by-Side Comparison**

• Conclusions
• Set of standard interface elements
  • e.g. Button, CheckBox, Slider

• Layouting

• Styling
  • Platform Look&Feel
  • Custom

• Application Window
  • Menu Bar, Tool Bar, Status Bar, etc

• Dialogs
  • Standard Dialogs
  • Base for Custom Dialogs
QtWidgets

- System
  - Graphics Buffers
- Qt Version
  - basically any
- Programming Languages
  - For Use
    - C++
    - (QML + JavaScript with QML registered widgets)
  - For Extending
    - C++

QtQuick.Controls

- System
  - OpenGL
- Qt Version
  - > 5.1
- Programming Languages
  - For Use
    - QML + JavaScript
  - For Extending
    - QML + JavaScript (composition)
    - C++ (custom rendering)
QtWidgets

- Horizontal/Vertical Box Layout, Grid Layout, Form Layout
  - Layouts "fill" parent widget

- Widgets provide:
  - Size Hints
  - Size Policies

- Developer can override size hint, set policy

QtQuick.Controls

- RowLayout, ColumnLayout, GridLayout, Anchors (relative positioning)
  - Layouts need to be explicitly sized or anchored

- Controls provide:
  - Implicit Size

- Developer can attach resize hints
**QtWidgets**

```cpp
QVBoxLayout *layout = new QVBoxLayout(this);
QPushButton *one = new QPushButton("One");
layout->addWidget(one);
QPushButton *two = new QPushButton("Two");
layout->addWidget(two);
QPushButton *three = new QPushButton("Three");
layout->addWidget(three);
```

**QtQuick.Controls**

```qml
ColumnLayout {
    anchors.fill: parent
    Button { text: "One" }
    Button { text: "Two" }
    Button { text: "Three" }
}
```
QtWidgets

- Platform Native Styling
- Qt Style Sheets (QSS)
- QStyle Plugins

QtQuick.Controls

- Platform Native Styling
- Style Component
  - Replace parts of the control
Styling - Example

QtWidgets

```cpp
QPushButton *button = new QPushButton("Click Me!", window);
button->setStyleSheet("QPushButton {background-color: white}");
```

QtQuick.Controls

```cpp
Button {
  x: 50; y: 50
  text: "Click Me!"

  style: ButtonStyle {
    background: Rectangle {
      color: "white"
      border.color: "#ABABAB"
    }
  }
}
```
QtWidgets

- QMainWindow
  - Menu Bar
  - Status Bar
  - Any number of Tool Bars
    - Adding actions results in Tool Buttons
  - Dock Widgets
  - Central Widget resized with window

QtQuick.Controls

- ApplicationWindow
  - Menu Bar
  - Status Bar
  - One Tool Bar
    - Create Tool Buttons, then associate action
    - Content item needs explicit resize handling
QtWidgets

- Standard Dialogs
  - Color, File, Font, MessageBox, Print, Progress, Wizard

- Custom Dialogs
  - Base type QDialog
    - Modal and Non-modal
  - QDialogButtonBox for platform correct button handling
    - access to individual buttons possible
  - accept/reject can be intercepted

QtQuick.Controls

- Standard Dialogs
  - Color, File, Font, Message

- Custom Dialogs
  - Base type Dialog
    - Modal and Non-modal
  - standardButtons for platform correct button handling
    - currently not access to buttons
  - All actions close the dialog

Side-by-Side Comparison
QtWidgets

- Qt Designer in QtCreator and stand-alone
  - Code generated by UIC
  - Used by delegation in custom classes

- Testing
  - QtTest for unit testing
  - Squish for UI testing

- Gammaray for runtime inspection

- C++ tools for debugging/analysis of custom code

QtQuick.Controls

- QtQuick Designer in QtCreator
  - Manipulates QML code directly
  - Use by manual editing of the same files

- Testing
  - QtTest for unit testing
  - Squish for UI testing

- Gammaray for runtime inspection

- QtCreator JS debugger/profiler

Side-by-Side Comparison
• The Question

• Side-by-Side Comparison

• Conclusions
• Both technologies viable for wide range of projects
• QtQuick.Controls not as complete yet but rapidly evolving
• Knowledge of types easily transferable
• Knowledge of behavior not always applicable