## Comparison **:** Choosing the appropriate technology for building a user interface is not always easy. It depends on several variables, such as other technologies used in a project or target platform constraints. To some extent, many items in the following table can be implemented using any of the three available technologies. The following table aims to help you choose the tool best suited for the job.

|  | **Qt Quick / Qt Quick Controls** | **Qt Widgets** | **Qt WebEngine** | **Comments** |
| --- | --- | --- | --- | --- |
| Used language(s) | QML/JS | C++ | HTML/CSS/JS |  |
| Native look'n'feel | IMG_256 | IMG_257 |  | Qt Widgets and Qt Quick Controls 1 integrate well to the underlying platform, providing a native look'n'feel on Windows, Linux, and macOS. |
| Custom look'n'feel | IMG_258 | IMG_259 | IMG_260 | Qt Widgets provide means for customization via style sheets, but Qt Quick is a better performing choice for user interfaces that do not aim to look native. |
| Fluid animated UIs | IMG_261 |  | IMG_262 | Qt Widgets do not scale well for animations. Qt Quick offers a convenient and natural way to implement animations in a declarative manner. |
| Touch screen | IMG_263 |  | IMG_264 | Qt Widgets often require a mouse cursor for good interaction, whereas Qt Quick only provides primitive building blocks that were designed with touch interaction in mind. The [WebView](https://doc.qt.io/qt-5/qml-qtwebview-webview.html) Qt Quick component has support for multi-touch gestures to interact with web content. |
| Standard industry widgets |  | IMG_265 |  | Qt Widgets provide all the bells and whistles, developed over two decades, needed for building standard industry type applications. Qt WebEngine Widgets provide widgets and additional classes to render and interact with web content. |
| Model/View programming | IMG_266 | IMG_267 |  | Qt Quick provides convenient views, but Qt Widgets provide more convenient and complete framework. In addition to Qt Quick views, Qt Quick Controls provide a [TableView](https://doc.qt.io/qt-5/qml-qtquick-tableview.html). |
| Rapid UI development | IMG_268 | IMG_269 | IMG_270 | Qt Quick is an excellent choice for rapid UI prototyping and development. |
| HW accelerated graphics | IMG_271 | IMG_272 | IMG_273 | Qt Widgets provide QGLWidget for rendering OpenGL graphics, and Qt WebEngine supports WebGL, but the OpenGL ES 2.0 or OpenGL 2.0 based [Qt Quick Scene Graph](https://doc.qt.io/qt-5/topics-graphics.html" \l "qt-quick-scene-graph) has proven to provide the best performance for UIs and for integrating with OpenGL content. |
| Graphical effects | IMG_274 |  |  | The particle system and shader effects available in Qt Quick are more flexible. Qt Widgets offer very little in this area. |
| Rich text processing | IMG_275 | IMG_276 |  | Qt Widgets currently provide the most comprehensive base for implementing text editors. Qt's rich text document classes can also be utilized in Qt Quick and Qt Quick Controls' [TextArea](https://doc.qt.io/qt-5/qml-qtquick-controls2-textarea.html), but may require some C++ implementation. |
| Existing web content |  |  | IMG_277 | Both Qt Quick and Qt Widgets provide components for presenting [simple rich text](https://doc.qt.io/qt-5/richtext-html-subset.html), but Qt WebEngine is the best choice for presenting full-blown web content. |