

# Qt

---

 Warning: The Symbian Signed program closed on January 1, 2014 and it is no longer possible to publish Qt apps (or other content) for Symbian and MeeGo on Nokia Store. It is possible to self-sign content for basic capabilities and continue to use existing DevCerts. For more information see this [blog](#).

Qt is the recommended native framework creating apps for Symbian and MeeGo Harmattan (Nokia N9). You will find that Qt greatly reduces your coding effort, through intuitive APIs that deliver more functionality from less code. When it comes to your app's UI, the declarative, JavaScript-like QML language within Qt Quick and tools such as Qt Quick Components enable you to create compelling UIs faster than you thought possible. And better still, you only need one core code-base to address the 180 million Qt-powered Symbian and Nokia N9 phones in consumers hands today. Qt enables code reuse that allows you to build and deploy versions of your apps for desktop platforms too. And you will find all of this is possible without compromising your apps performance because Qt is a framework, not a runtime.

This article provides a getting started tutorial, along with links to related documentation, examples, and other resources.

## Documentation

---

In addition to the API documentation in Qt Creator (**Help > Qt Reference Documentation**), see the other relevant documentation resources:

- [Qt SDK 1.2 Documentation](#) (doc.qt.digia.com) - this is the latest release compatible with Symbian and MeeGo Harmattan devices
- [MeeGo 1.2 Harmattan Developer Documentation](#)
- [Category:Qt](#) (Wiki)
  - [Platform Security](#)

## Download

---

- [QtSDK - Offline for Windows x86 v1.2.1](#)
- [QtSDK - Offline for OSX x86 v1.2.1](#)
- [QtSDK - Offline for Linux x64 v1.2.1](#)
- [QtSDK - Offline for Linux x86 v1.2.1](#)

## Code Examples

---

- [Portal:Qt Code Examples](#) (Community contributed articles with code)

## Learn Qt

---

Learning how to code Qt applications is easy, no matter which programming background you have. Qt Quick consists of a declarative QML language and JavaScript. QML syntax is easy and especially handy when developing UIs and using animations. Together with JavaScript for application logic it's fast to develop complete applications. However, in many cases C++ may be a better option, C++ programmers will find Qt it fast to learn as there are few Qt-specific idioms and coding conventions. To learn more about Qt Quick principles go to **Help > Qt Reference Documentation > What is Qt > Qt Quick**. To achieve a platform look and feel in Symbian and Harmattan, we recommend using Qt Quick Components (Components for Symbian / Components for Harmattan). Read more on the available components in the SDK documentation by going to **Help > Qt Quick Components for Symbian**.

There are also heaps of useful resources at <http://qt.digia.com/>

## Use Qt Reference Documentation to find information on available the APIs

When you are ready to start developing an application, API documentation is available for you in Qt Creator: go to **Help > Qt Reference Documentation** for comprehensive documentation, including tutorials and example applications.

## Examine code examples

A full set of Qt cross-platform examples is available in the <QtSDK\_install\_path>\Examples directory. All example applications come with a full source that you can examine and utilise in your own applications.

In addition, there are a large number of community-contributed articles with code: [Portal:Qt Code Examples](#)

## Use mobile features

The Qt Mobility API provides a set of interfaces — such as Messaging, Contacts, and Location — to enable mobile use cases. The Mobility API is available in two flavours: as QML bindings, for easy access from a Qt Quick application, and as C++ APIs. For in-depth API documentation, go to **Help > Qt Mobility Project Reference Documentation** in Qt Creator. You can also check out the online documentation.

## Expand with platform features

Even though the Qt APIs offer a comprehensive set of features for richly functional applications targeted at the Symbian and MeeGo 1.2 Harmattan platforms, in some applications there may be a need for additional platform features that are not part of Qt or the Qt Mobility API. In these cases you have the option to use Symbian and Harmattan platform APIs from Qt Creator. Symbian C++ API documentation is available in **Help > Symbian Reference Documentation** for Qt. For Harmattan documentation, go to the [MeeGo 1.2 Harmattan Developer Library](#).

## Community support

Visit the Qt [discussion boards](#) for community support.