

## Course Overview

Qt is a multiplatform C++ application development framework. It has become the emerging standard development environment for IT professionals who want to write a single source-tree, port it to multiple platforms with a simple recompile, integrate it easily with Motif and ActiveX, make it database-capable, globalize it, give it a native look and feel, and run it at native speed. After completion you will have: A knowledge of the capabilities & limitations of Qt; A knowledge of multi platform development using Qt; Experience programming with Qt.

## Pre-Requisites

Very good knowledge of the C++ programming language. Awareness of atleast on UI toolkit is a plus, but not required, Should be comfortable using a GNU/Linux distribution. Knowledge of OpenGL would be a plus (only for learning the Qt/OpenGL module)

## Target Audience

Programmers developing applications for desktops, embedded devices and/or targeting multiple platforms - Windows, Unix and Mac will greatly benefit from this training.

## Course Outline

Introduction to Qt

- Survey of GUI toolkits; Technical and product history of Qt, Trolltech.
- Dual-license of Qt

Installation

- Getting Qt sources, Compiling & Installing Qt, "Hello World" program in Qt

Introduction to Signals and Slots

- Making, Breaking a connection; Creating signals and slots

Introduction to UI programming in Qt

- The QWidget class; the QMainWindow class; Qt Designer
- User actions: QAction/QToolButton/QMenu/QToolBar; Exercise: Text Editor

Containers, Types, and Macros.

- QList, QMap, QStack & Interacting containers

Common Widgets

- QPushButton, QCheckBox, QRadioButton,
- QLabel, QLineEdit, QSpinBox, QGroupBox, QSlider & QProgressBar

Common Dialogs

- QFileDialog, QInputDialog & QColorDialog
- QMessageBox ; QProgressDialog

Custom Widgets

- Form Widgets; Custom controls

Layouts

- Vertical and Horizontal layouts; Grid layouts; Form layouts
- Custom layouts; Drawing & Printing (Arthur Framework)
- Drawing framework: QPainter, QPaintDevice, PaintEngine
- Helper classes: QRect, QPoint, QPen, QRush, Gradient & QPainterPath
- Drawing on widgets; Drawing on pixmap
- Printing; Transformations: QMatrix, QTransform

Handling Events in Qt

- Event model in Qt. (QEvent and subclasses, QObject::event()) method)
- Event handlers in Qt (paintEvent, mousePressEvent, mouseReleaseEvent)
- Event filters; Signal/Slot internals (QMetaObject and friends)

The Model-View Framework (Interview)

- Introduction to MVC Design Pattern
- QAbstractItemModel – Qt's Model class
- QAbstractItemView – Qt's View class
- QAbstractItemDelegate – Qt's Delegate class
- QModelIndex – Qt's data pointer into model
- Built in models: QDirModel, QStringListModel, QFileSystemModel
- Built in views: QListView, QTreeView, QColumnView, QTableView
- Built in item views: QListWidget, QTreeView, QTableWidget
- Writing custom models and views

Graphics View Framework

- QGraphicsScene, QGraphicsView & QGraphicsItem
- OTS Items: QGraphicsLineItem, QGraphicsRectItem, QGraphicsEllipseItem
- Custom items; Transformations and Interactive graphics

Files, Streams

- IO Device framework: QIODevice; Built in IO Devices: QFile, QBuffer
- Stream classes: QTextStream & QDataStream
- Buffer classes: QByteArray, QString

Help System (Qt 4.4 module)

- How to create compressed help files in Qt
- Using compressed help files in Qt assistant
- Showing help in your applications using QHelpEngine & friends

Multimedia (Phonon)

- How to play audio and video files in Qt

HTML Rendering (WebKit)

- Introduction to WebKit project; QWebView – the Browser control in Qt

Making your applications scriptable (QScript)

- Basic concepts: What does a scriptable application mean
- Evaluating scripts using QScriptEngine
- Accessing Qt/C++ objects in the script environment
- Accessing variables in the script within Qt/C++
- Designing objects for scriptability

XML

- DOM and SAX; Parsing XML files; Authoring XML files

OpenGL

- The QGLWidget class; Using OpenGL in graphics view
- Using OpenGL in your widgets

Doing Things in Parallel. (Threads and QProcess)

- QThread, QMutex and QSemaphore; - QProcess

Plugin system

- QPlugin and friend classes; Creating Qt Designer plugins
- Architecting applications to allow extensions via plugins
  - How to load and use plugins
- Using QLibrary to dynamically load libraries

Networking

- Understanding QAbstractSocket
- TCP/IP communication using QTcpSocket & QTcpServer
- UDP communication using QUDPsocket
- FTP / HTTP Transactions

Introduction to Third-Party Libraries

- Qwt – Graphing Library
- GCF – Components Framework Library (similar to Microsoft's COM & KParts)

**Course Duration:** Four Days: 9 am - 6.00 pm

**Course Fee Rs. 15,000/-**

(Plus Service Tax as applicable)