Nokia Certified Qt Developer

LAYTRONICS SOLUTIONS Enabling Smart Solutions

Nokia Certified Qt Specialist

"Changing the way we think and learn"

Qt Training Details

Qt is a multiplatform C++ application development framework. Qt allows you to write advanced applications and UIs once, and deploy them across desktop and embedded operating systems without rewriting the source code saving time and development cost. It provides a broad set of customizable widgets, graphics canvas, style engine and more that you need to build modern user interfaces.

Course duration

5 Days (9-00 AM to 5-00 M)

Course delivery method

- Presentation and Demonstration (<50%)
- Hands-On Development (>50%)
- There will be review paper after every day

Hand-on Platform

All the examples will be run on Linux Desktop with Qt SDK and Qt Creator. It is recommended to have the one desktop for each participant. Please arrange the desktop pc for each participant and install the Qt 4.7.x from http://qt.nokia.com/. Also we can test few samples on ARM Board as demonstration purpose.

Target audience

Qt developers

Prerequisites

C++ Programming
Basics of Object Oriented programming (Class, Inheritance, Polymorphism, Virtual functions)
Basics of Qt (Does not hurt☺)

Course level

The course level is *Basic/Intermediate (2-3 Days)* and *Intermediate/*Advanced *(2-3 Days)*

Course Objectives

Learn the Fundamentals and Advanced topics of Qt Programming

Nokia Certified Qt Developer

Nokia Certified Qt Specialist

Qt Essentials and Advanced course contents

1. **Introduction**
   - GUI Programming Fundamentals
   - Qt Overview and Status

2. **Qt Essentials**
   - Qt Modules
   - QT Build mechanism
   - Qt Tools(Qt Creator, Qt Assistant, Qt Designer)
   - Examples/Usecases

3. **Basic Building of Qt Application**
   - QtCreator and First Application
   - Basic Data types
   - QCoreApplication, QApplicaiton and internals
   - Simple UI application
   - Native APIs and Cross-Platform APIs
   - Examples/Usecases

4. **Memory Management**
   - Qt Object Model
   - Parent/Child Relationship
   - Object Hierarchy
   - Heap and Stack allocation of Objects
   - Garbage Collection
   - QPointer, QScope Pointer etc
   - Examples/Usecases

5. **Signals and Slots**
   - Meta-Objectsand QObjects
   - MOC Compiler and MOC File Generation
   - Why Signals and Slots?
   - Signal & Slots Communication & Delivery
   - Data passing
   - SignalMapper
   - Benefits of Signals and Slots
   - Examples/Usecases

6. **UI Designing with Qt Designer**
   - UI Design with Designer and internals
   - Designer options (e.g size policy etc)
   - Creating own customized Dialogs, MainWindow, Widgets
   - Menus, Action and MainWindow

- Examples/Usecases

7. **Layout management**
   - Layout concepts
   - Laying - Rows & Columns
   - Vertical and Horizontal Layout
   - Grid Layout
   - Form Layout
   - Nested Layout managers
   - Geometry Management
   - Examples/Usecases

8. **Event Management**
   - Event flow and Handling
   - Event types (e.g keyboard, Mouse, Timer etc)
   - Event model and Event loop
   - Event Filters
   - Custom Event Mechanism
   - Timers in Qt
   - Examples/Usecases

9. **Widgets and Dialogs**
   - Introduction Widgets
   - Writing custom Widgets
   - QPainter and capabilities
   - Event Processing and Paint Engine
   - Introduction to Graphics View Architecture
   - Various Dialog types
   - Examples/Usecases
   -

10. **Plugin Architecture**
    - Dynamic library and Usage
    - Static Library and Usage
    - Plugin concepts
    - QtDesigner plugin

11. **I/O and Files**
    - QIO Device
    - QTextStream and QDataStream
    - Reading and Writing files
    - Examples/Usecases

12. **Process and Threads**
    - QProcess
    - QThreads
    - Threads and Event loop
    - Examples/Usecases

13. **Inter-process communication (IPC)**
    - Synchronization Objects - Mutexes & Semaphores etc
    - Shared Memory
    - Message Queue
    - Network Module
    - QT DBUS Architecture
    - Examples/Usecases

14. **Databases**
    - Connecting to DB
    - Query, Viewing
    - Editing records
    - Tabular Data Presentation

15. **Internationalization**
    - Multi language Support
    - Translation aware Applications
    - Dynamic Language Switching

16. **Qt Model View Framework**
    - MVC Design Pattern(Model / View concepts)
    - Qt's Model class and Built in models
    - Qt's View class and built in views
    - Qt's Delegate class
    - Built in item views: QListWidget, QTreeWidget, QtableWidget
    - Custom models and views

17. **Qt Painter and Image Management**
    - QPainter and capabilities
    - Image handling in Qt.
    - Classes for image Handling (QImage, QPixMap, QBitMap etc)
    - I/O with Image Files
    - Pixel Handling
    - Image Transformations

18. **XML**
    - SAX Parser
    - DOM Parser
    - Qt XML Parser
    - Reading the XML files
    - Creating XML File Dynamically
    - Storing the XML Files