Qt Essentials - Fundamentals of Qt Module

Training Course

Visit us at `http://qt.digia.com`

Produced by Digia Plc.

Material based on Qt 5.0, created on September 27, 2012

Digia Plc.

- The Story of Qt
- Developing a Hello World Application
- Hello World using Qt Creator
- Practical Tips for Developers

Module Learning Objectives

- Learn ...
  - ... about the history of Qt
  - ... about Qt's ecosystem
  - ... a high-level overview of Qt
  - ... how to create first hello world program
  - ... build and run a program cross platform
  - ... to use Qt Creator IDE
  - ... some practical tips for developing with Qt

digia

Meet Qt

- **Qt Development Frameworks founded in 1994**
  - Trolltech acquired by Nokia in 2008
  - Qt Commercial business acquired by Digia in 2011
  - Qt business acquired by Digia from Nokia in 2012
  - Trusted by over 6,500 companies worldwide
- **Qt: a cross-platform application and UI framework**
  - For desktop, mobile and embedded development
  - Used by more than 350,000 commercial and open source developers
  - Backed by Qt consulting, support and training

See Qt Services

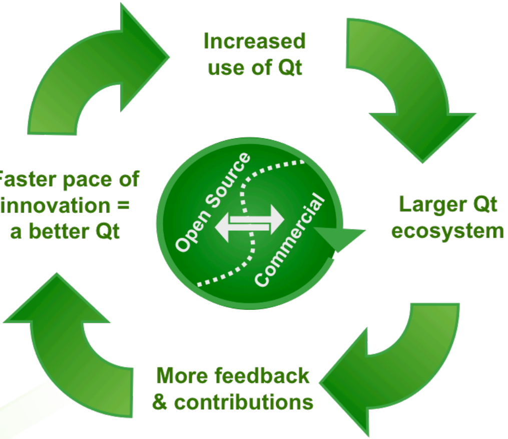
Qt is used everywhere

From embedded devices to desktop applications

By companies from many industries

Alcatel·Lucent AMD Google Canon
BT DAIMLER
EPSON
esa freescale semiconductor HITACHI Inspire the Next
GENERAL DYNAMICS C4 Systems
Mentor Graphics PGS HONDA The Power of Dreams
MITSUBISHI HEAVY INDUSTRIES, LTD.
NORTEL NETWORKS ORACLE PHILIPS
Pioneer skype Raytheon
Schlumberger ROHDE&SCHWARZ SCANIA
SENNHEISER SHARP
xerox vodafone
VOLVO THOMSON REUTERS

The Qt virtuous cycle


Increased use of Qt
Larger Qt ecosystem
More feedback & contributions
Faster pace of innovation = a better Qt
Open Source
Commercial

See Qt Licensing

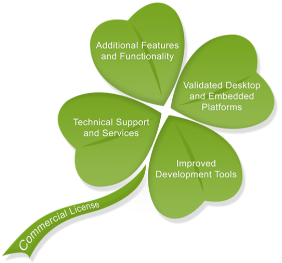
Why Qt

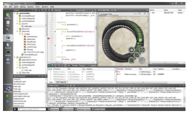
- Write code once to target multiple platforms
- Produce compact, high-performance applications
- Focus on innovation, not infrastructure coding
- Choose the license that fits you
  - Commercial, LGPL or GPL
- Count on professional services, support and training
- Take part in an active Qt ecosystem

15 years of customer success and community growth

Why Qt Commercial

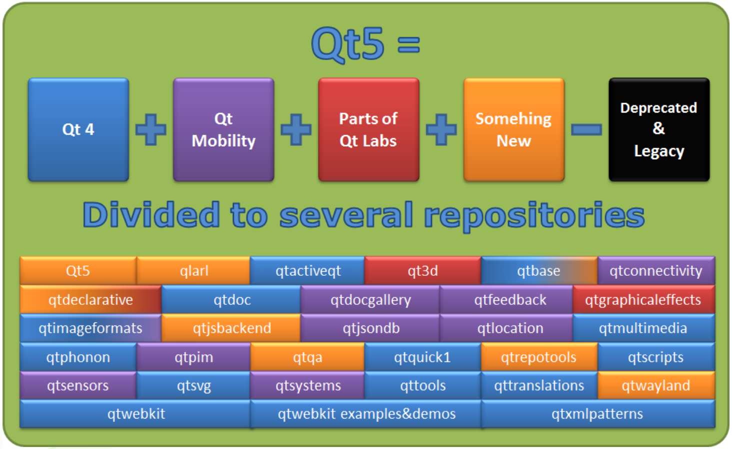
- Improved development tools for increased productivity and tangible cost savings
- Qt Commercial SDK
  - All Qt Commercial libraries and tools
  - Additional tools and components
- Qt Creator Embedded Target
  - Deploy directly to embedded Linux target
- RTOS toolchain integration
- Visual Studio Add-On


Additional Features and Functionality
Validated Desktop and Embedded Platforms
Technical Support and Services
Improved Development Tools
Commercial License



Qt5

- Awesome graphics capabilities
  - OpenGL as a standard feature of user interfaces
  - Shader-based graphics effects in QtQuick 2
- New modular structure
  - Qt Essential modules available on all platforms
  - Add-on modules provide additional or platform-specific functionality
- Developer productivity and flexibility
  - More web-like development with QtQuick 2, V8 JvaScript engine, and Qt JSON DB
- Cross-platform portability
  - Qt Platform Abstraction (QPA) replaces QWS and platform ports



Qt5 =

Qt 4 + Qt Mobility + Parts of Qt Labs + Somehing New − Deprecated & Legacy

Divided to several repositories

Qt5	qlarl	qtactiveqt	qt3d	qtbase	qtconnectivity
qtdeclarative	qtdoc	qtdocgallery	qtfeedback	qtgraphicaleffects	
qtimageformats	qtjsbackend	qtjsondb	qtlocation	qtmultimedia	
qtphonon	qtpim	qtqa	qtquick1	qtrepotools	qtscripts
qtsensors	qtsvg	qtsystems	qttools	qttranslations	qtwayland
qtwebkit	qtwebkit examples&demos	qtxmlpatterns			

Qt5 Highlights

- QtCore
  - JSON parser and speed optimized binary format for JSON
  - Compile time checked signal/slot connection syntax
  - New plugin loader - no need to load plugnis to see what they implement
  - Re-written QMap for optimized performance
- QtGui
  - Printing and widgets moved into their own libs
  - Platform ports based on QPA
  - QApplication split into QApplication and QGuiApplication
  - QWindow to reprsent a top-level surface
  - Touch improvements (device capabilities like pressure)
- QtQuick 2
  - Support for touch gestures
  - Locale API to QML
  - Particle system
  - RTL support

- Let's have a look at the QtDemo Application
- Comes with every Qt installation

Technology	**Demo**
Painting	*Demonstrations/Path Stroking*
Widgets	*Demonstrations/Books*
Widgets	*Demonstrations/TextEdit*
Graphics View	*Demonstrations/40.000 Chips*
OpenGL	*Demonstrations/Boxes*
WebKit	*Demonstrations/Browser*

Qt Examples and Demos

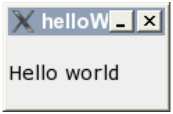
Qt is supplied with a number of example applications and demonstrations that have been written to provide developers with examples of the Qt API in use, highlight good programming practice, and showcase features found in each of Qt's core technologies.

The example and demo launcher can be used to explore the different categories available. It provides an overview of each example, lets you view the documentation in Qt Assistant, and is able to launch examples and demos.

Documentation for examples can be found in the Tutorials and Examples section of the Qt documentation.

``Hello World'' in Qt

```
// main.cpp

#include <QtWidgets>

int main(int argc, char *argv[])
{
  QApplication app(argc, argv);
  QPushButton button("Hello world");
  button.show();
  return app.exec();
}
```



- Program consists of
  - `main.cpp` - application code
  - `helloworld.pro` - project file

Demo fundamentals/ex-helloworld

Project File - `helloworld.pro`

- `helloworld.pro` file
  - lists source and header files
  - provides project configuration

```
# File: helloworld.pro
SOURCES  = main.cpp
HEADERS +=          # No headers used
QT       = core gui widgets
```

- Assignment to variables
  - Possible operators `=, +=, -=`

See qmake tutorial Documentation

Using qmake

- `qmake` tool
  - Creates cross-platform make-files
- Build project using qmake

```
cd helloworld
qmake helloworld.pro # creates Makefile
make                 # compiles and links application
./helloworld         # executes application
```

- Tip: `qmake -project`
  - Creates default project file based on directory content

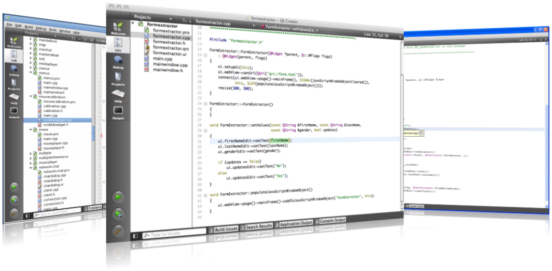
See qmake Manual Documentation

Qt Creator does it all for you

- The Story of Qt
- Developing a Hello World Application
- **Hello World using Qt Creator**
- Practical Tips for Developers

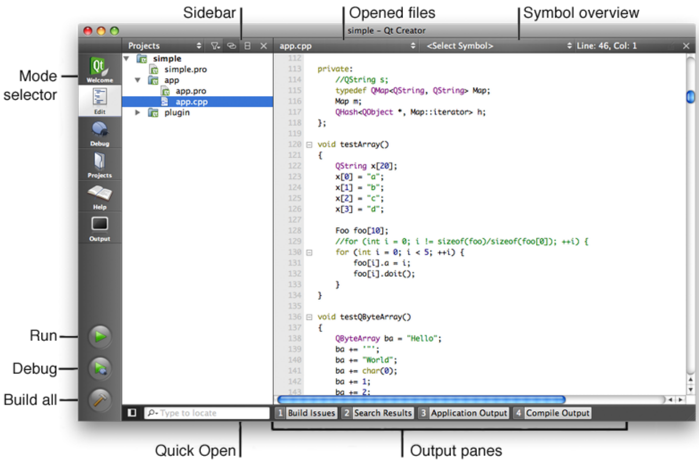
QtCreator IDE

- Advanced C++ code editor
- Integrated GUI layout and forms designer
- Project and build management tools
- Integrated, context-sensitive help system
- Visual debugger
- Rapid code navigation tools
- Supports multiple platforms



```
#include "formextractor.h"

FormExtractor::FormExtractor(QWidget *parent, Qt::WFlags flags)
    : QWidget(parent, flags)
{
    ui.setupUi(this);
    ui.webView->setUrl(QUrl("qrc:/form.html"));
    connect(ui.webView->page()->mainFrame(), SIGNAL(javaScriptWindowObjectCleared()),
            this, SLOT(populateJavaScriptWindowObject()));
    resize(300, 300);
}

FormExtractor::~FormExtractor()
{
}

void FormExtractor::setValues(const QString &firstName, const QString &lastName,
                              const QString &gender, bool updates)
{
    ui.firstNameEdit->setText(firstName);
    ui.lastNameEdit->setText(lastName);
    ui.genderEdit->setText(gender);

    if (updates == false)
        ui.updatesEdit->setText("No");
    else
        ui.updatesEdit->setText("Yes");
}

void FormExtractor::populateJavaScriptWindowObject()
{
    ui.webView->page()->mainFrame()->addToJavaScriptWindowObject("formExtractor", this);
}
```

Overview of Qt Creator Components

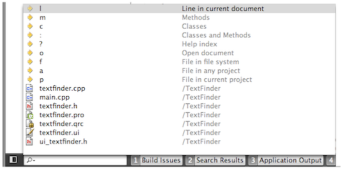


```

private:
    //QString s;
    typedef QMap<QString, QString> Map;
    Map m;
    QHash<QObject *, Map::iterator> h;
};

void testArray()
{
    QString x[20];
    x[0] = "a";
    x[1] = "b";
    x[2] = "c";
    x[3] = "d";

    Foo foo[10];
    //for (int i = 0; i != sizeof(foo)/sizeof(foo[0]); ++i) {
    for (int i = 0; i < 5; ++i) {
        foo[i].a = i;
        foo[i].doit();
    }
}

void testQByteArray()
{
    QByteArray ba = "Hello";
    ba += "";
    ba += "World";
    ba += char(0);
    ba += 1;
    ba += 2;
```

digia

Finding Code -Locator

- Click on Locator or press Ctrl+K (Mac OS X: Cmd+K)
- Type in the file name
- Press Return

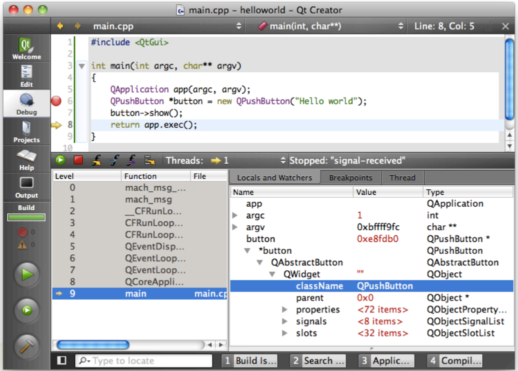


l	Line in current document
m	Methods
c	Classes
:	Classes and Methods
?	Help index
o	Open document
f	File in file system
a	File in any project
p	File in current project
textfinder.cpp	/TextFinder
main.cpp	/TextFinder
textfinder.h	/TextFinder
textfinder.pro	/TextFinder
textfinder.qrc	/TextFinder
textfinder.ui	/TextFinder
ui_textfinder.h	/TextFinder

Locator Prefixes

- **: <class name>** - Go to a symbol definition
- **l <line number>** - Go to a line in the current document
- **? <help topic>** - Go to a help topic
- **o <open document>** - Go to an opened document

- Debug – > Start Debugging (or F5)



```
#include <QtGui>

int main(int argc, char** argv)
{
    QApplication app(argc, argv);
    QPushButton *button = new QPushButton("Hello world");
    button->show();
    return app.exec();
}
```

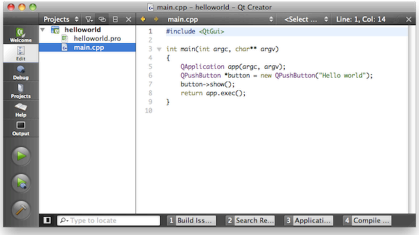
Level	Function	File
0	mach_msg_...	
1	mach_msg	
2	__CFRunLo...	
3	CFRunLoop...	
4	CFRunLoop...	
5	QEventDisp...	
6	QEventLoop...	
7	QEventLoop...	
8	QCoreAppli...	
9	main	main.c

Name	Value	Type
app		QApplication
argc	1	int
argv	0xbffff9fc	char **
button	0xe8fdb0	QPushButton *
*button		QPushButton
QAbstractButton		QAbstractButton
QWidget	""	QObject
className	QPushButton	
parent	0x0	QObject *
properties	<72 items>	QObjectProperty...
signals	<8 items>	QObjectSignalList
slots	<32 items>	QObjectSlotList

Qt Creator Demo "Hello World"

What we'll show:

- Creation
  of an empty Qt project
- Adding
  the `main.cpp` source file
- Writing of
  the Qt Hello World Code
  - Showing Locator Features
- Running the application
- Debugging the application
  - Looking up the `text` property of our button



```
#include <QtGui>

int main(int argc, char** argv)
{
    QApplication app(argc, argv);
    QPushButton *button = new QPushButton("Hello world");
    button->show();
    return app.exec();
}
```

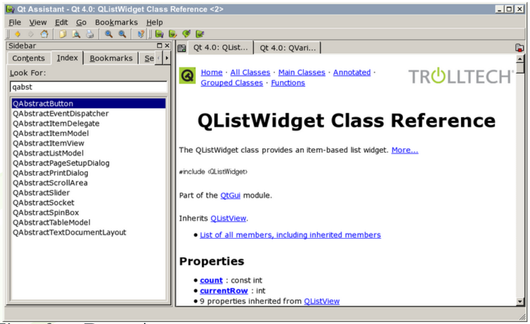
- The Story of Qt
- Developing a Hello World Application
- Hello World using Qt Creator
- Practical Tips for Developers

How much C++ do you need to know?

- Objects and classes
  - Declaring a class, inheritance, calling member functions etc.
- Polymorphism
  - That is virtual methods
- Operator overloading
- Templates
  - For the container classes only
- No ...
  - ... RTTI
  - ... sophisticated templates
  - ... exceptions thrown
  - ...

Qt Documentation

- Reference Documentation
  - All classes documented
  - Contains tons of examples
- Collection of Howto's and Overviews
- A set of Tutorials for Learners


Qt Assistant - Qt 4.0: QListWidget Class Reference <2>
File View Edit Go Bookmarks Help
Sidebar
Contents Index Bookmarks Se
Look For:
qabst
Qt 4.0: QList... Qt 4.0: QVari...
Home · All Classes · Main Classes · Annotated · Grouped Classes · Functions
TROLLTECH

QAbstractButton
QAbstractEventDispatcher
QAbstractItemDelegate
QAbstractItemModel
QAbstractItemView
QAbstractListModel
QAbstractPageSetupDialog
QAbstractPrintDialog
QAbstractScrollArea
QAbstractSlider
QAbstractSocket
QAbstractSpinBox
QAbstractTableModel
QAbstractTextDocumentLayout

QListWidget Class Reference

The QListWidget class provides an item-based list widget. More...

#include <QListWidget>

Part of the QtGui module.

Inherits QListView.

- List of all members, including inherited members

Properties

- **count** : const int
- **currentRow** : int
- 9 properties inherited from QListView

Finding the Answers

- Documentation in Qt Assistant (or QtCreator)
- Qt's examples: `$QTDIR/examples`
- Qt developer network: `http://qt-project.org/`
- Qt Centre Forum: *`http://www.qtcentre.org/`*
- KDE project source code
  - `http://lxr.kde.org/` *(cross-referenced).*
- Online communities

  `http://qt-project.org/wiki/OnlineCommunities`

Use the source!

Qt's source code is easy to read, and can answer questions the reference manual cannot answer!

Modules and Include files

- Qt Modules
  - QtCore, QtGui, QtWidgets, QtXml, QtSql, QtNetwork, QtTest ...

    See Qt Modules Documentation
- Enable Qt Module in qmake `.pro` file:
  - `QT += network`
- Default: qmake projects use QtCore and QtGui
  - Any Qt class has a header file.

```
#include <QLabel>
#include <QtWidgets/QLabel>
```

  - Any Qt Module has a header file.

```
#include <QtGui>
```

Includes and Compilation Time

Module includes

```
#include <QtGui>
```

- Precompiled header and the compiler
  - If **not** supported may add extra compile time
  - If supported may speed up compilation
  - Supported on: Windows, Mac OS X, Unix

See qmake precompiled headers Documentation

Class includes

```
#include <QLabel>
```

- Reduce compilation time
  - Use class includes (#include <QLabel>)
  - Forward declarations (`class QLabel;`)

Place module includes before other includes.

Questions And Answers

- What is Qt?
- Which code lines do you need for a minimal Qt application?
- What is a .pro file?
- What is `qmake`, and when is it a good idea to use it?
- What is a Qt module and how to enable it in your project?
- How can you include a `QLabel` from the `QtGui` module?
- Name places where you can find answers about Qt problems

Questions And Answers

- What is Qt?
- Which code lines do you need for a minimal Qt application?
- What is a .pro file?
- What is `qmake`, and when is it a good idea to use it?
- What is a Qt module and how to enable it in your project?
- How can you include a `QLabel` from the `QtGui` module?
- Name places where you can find answers about Qt problems

Questions And Answers

- What is Qt?
- Which code lines do you need for a minimal Qt application?
- What is a .pro file?
- What is `qmake`, and when is it a good idea to use it?
- What is a Qt module and how to enable it in your project?
- How can you include a `QLabel` from the `QtGui` module?
- Name places where you can find answers about Qt problems

Questions And Answers

- What is Qt?
- Which code lines do you need for a minimal Qt application?
- What is a .pro file?
- What is `qmake`, and when is it a good idea to use it?
- What is a Qt module and how to enable it in your project?
- How can you include a `QLabel` from the `QtGui` module?
- Name places where you can find answers about Qt problems

Questions And Answers

- What is Qt?
- Which code lines do you need for a minimal Qt application?
- What is a .pro file?
- What is `qmake`, and when is it a good idea to use it?
- What is a Qt module and how to enable it in your project?
- How can you include a `QLabel` from the `QtGui` module?
- Name places where you can find answers about Qt problems

Questions And Answers

- What is Qt?
- Which code lines do you need for a minimal Qt application?
- What is a .pro file?
- What is `qmake`, and when is it a good idea to use it?
- What is a Qt module and how to enable it in your project?
- How can you include a `QLabel` from the `QtGui` module?
- Name places where you can find answers about Qt problems

Questions And Answers

- What is Qt?
- Which code lines do you need for a minimal Qt application?
- What is a .pro file?
- What is `qmake`, and when is it a good idea to use it?
- What is a Qt module and how to enable it in your project?
- How can you include a `QLabel` from the `QtGui` module?
- Name places where you can find answers about Qt problems