entwickler.press

Qt-Entwicklung

für Symbian, Android und Desktop

Tam Hanna

Tam Hanna

Qt-Entwicklung

für Symbian, Android und Desktop

entwickler.press

Tam Hanna
Qt-Entwicklung für Symbian, Android und Desktop
ISBN: 978-3-86802-249-0

Ein Imprint der Software & Support Media GmbH

Bibliografische Information Der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

Ihr Kontakt zum Verlag und Lektorat:
Software & Support Media GmbH
entwickler.press
Geleitsstr. 14
60599 Frankfurt am Main
Tel.: +49 (0)69 630089-0
Fax: +49 (0)69 930089-89
lektorat@entwickler-press.de
http://www.entwickler-press.de

Lektorat: Sebastian Burkart
Korrektorat: Redaktion ALUAN Köln
Satz: Dominique Kalbassi
Belichtung, Druck & Bindung: M.P. Media-Print Informationstechnologie GmbH, Paderborn

Inhaltsverzeichnis

Vorwort

Korrekturen herunterladen – JETZT

Qt entwickelt sich mit rasender Geschwindigkeit – wenn ein Lehrbuch zum Thema gedruckt ist, ist es schon schwer veraltet. Aus diesem Grund gibt es zu diesem Buch aktuelle Korrekturen im PDF-Format, die unter *www.entwickler-press.de/qt* heruntergeladen werden können. Bitte werfen Sie regelmäßig einen Blick auf die Buchseite um keine Aktualisierung zu verpassen – es wird ihnen viel Aufwand ersparen.

Sollten Sie selbst etwas Seltsames bemerken, melden Sie sich bitte unter *tamhan@tamoggemon.com*. Wer etwas Produktives beiträgt, ist nicht nur ein guter Mensch, er bekommt zusätzlich eine kostenlose Lizenz eines Programms meines Unternehmens für sein Telefon – sozial sein lohnt sich also!

Über den Autor

Meine Wenigkeit befasst sich seit der Zeit des mittlerweile legendären Palm IIIc mit Mobilcomputertechnik – also seit der guten alten Zeit, wo Displays noch groß waren, man einen Stift hatte und Anwendungen 10 bis 50 Dollar kosteten ☺.

Neben der Programmierung von Anwendungen und Spielen für Handcomputer befasse ich mich mit dem Schreiben von Fachartikeln, dem Verfassen von Fachbüchern und einer Gruppe von Webseiten zum Thema Handcomputer.

Danksagung

Zu guter Letzt sei an dieser Stelle einigen Individuen gedankt, deren Hilfe dieses Buch überhaupt erst möglich gemacht hat.

Als Erstes gilt mein Dank Ihnen, dem Käufer dieses Buchs. Erst durch ihr Vertrauen macht meine Arbeit Sinn – ich hoffe, dass sie das Buch zufriedenstellen wird.

Durch lautes Quengeln beansprucht meine Lebensgefährtin auch diesmal den zweiten Platz in dieser illustren Aufzählung. Hätte Dr. Doris Maria Kohrs ihren Aufmerksamkeitsbedarf nicht radikal gedrosselt, wäre dieses Buch nie fertiggeworden.

Für die Fertigstellung gebührt ebensoviel Dank dem Team von entwickler.press: Herrn Burkart dafür, dass er die diversen Versionen meiner Texte zusammensammelte und „menschliches CVS“ spielte. Frau Kalbassi dafür, dass sie wahre Meisterleistungen im Satz vollbrachte und meine PDF-Korrekturen mühevoll und ohne Murren einpflegte. Zu guter Letzt gebührt auch Herrn Wießeckel Dank dafür, dass er mir die Weiterverwendung der bei ihm erschienenen Texte erlaubte.

Dank gebührt auch einigen Personen bei Nokia, die mir im Rahmen meiner Arbeit sehr behilflich waren.

Zu guter Letzt danke ich allen Kunden meines Unternehmens. Ohne euer Vertrauen würde es heute keine Tamoggemon Limited geben, und ohne Tamoggemon Limited auch kein Buch über Qt. Danke euch allen!

1 Einleitung

1.1 Warum Qt?

Auch wenn man es als Programmierer nicht gerne sieht: Hardwarehersteller profitieren von proprietären Plattformen. Nicht nur im Handcomputermarkt übersteigt der Preis der verwendeten Programme oftmals den der Hardware – und Software wird so zu einem wichtigen Kundenbindungsinstrument.

Programmiersprachen wie C oder C++ erlauben seit langer Zeit, Algorithmen und „Business Intelligence" plattformunabhängig zu gestalten. Leider gilt dies nicht für die Benutzerschnittstelle: Jeder Betriebssystemhersteller kocht in diesem Bereich sein eigenes, zu allen anderen Systemen inkompatibles „Süppchen".

Seit langer Zeit kämpfen Softwarehäuser gegen diesen Trend an. Es ist beispielsweise bekannt, dass diverse Mobilcomputerhäuser wie TealPoint und DataViz eigene Cross-Platform-Toolkits verwenden – leider ist deren Wartung aufwendig und teuer.

Trolltech entstand aus der Idee, diese Arbeit selbst zu übernehmen und an andere Firmen zu verkaufen. Mit ausreichend Partnern würden die Kosten pro Partner immer geringer werden und am Ende gegen Null konvergieren …der Rest ist Geschichte bzw. Handlung dieses Buchs.

1.2 Für wen ist dieses Buch?

Wie bereits angedeutet, werden Qt-Anwendungen in C++ programmiert. Aus diesem Grund sind grundlegende C++-Kenntnisse zwingend erforderlich, um von diesem Buch profitieren zu können.

Begriffe wie Klassen, Vererbung und Pointer sollten für Sie kein Problem darstellen – komplexere Themen wie Templates oder die Standard-Bibliothek finden, wenn überhaupt, nur am Rande Erwähnung.

In keinem Fall sind Vorkenntnisse in Qt erforderlich. Der erste Teil des Buchs zeigt alle für die praktische Anwendungsentwicklung relevanten Teile des Toolkits im Überblick – auch wenn Sie von Qt keine Ahnung haben und Mobiltelefone abgrundtief hassen, sind Sie hier richtig aufgehoben.

Auch die technischen Voraussetzungen sind eher moderat. Jede beliebige Windows-XP-Maschine mit mehr als 1GB RAM und 5 bis 10 GB freiem Festwertspeicher ist brauchbar – der Autor verwendet sowohl ein MSI-Wind-U100-Netbook als auch eine Zweikern-Workstation. Ein zweiter Monitor mit hoher Auflösung ist hilfreich.

Die mobilcomputerspezifischen Kapitel kann man logischerweise am besten nutzen, wenn man ein kompatibles Telefon besitzt. Mehr dazu in den jeweiligen Einleitungen.

Linux- und Mac-OS-Anwender werden im ersten Teil keine Probleme haben. Qt Creator rennt auch auf ihren Maschinen, der Entwicklung steht also nichts im Weg. Im zweiten Teil sieht die Lage weniger gut aus: Da viele Mobilcomputerhersteller ihre SDKs immer noch nur unter Windows anbieten, schaut man als Nicht-Windows-User in die Röhre.

1.3 Wie soll man dieses Buch lesen?

Die ersten Kapitel des ihnen vorliegenden Buchs bauen aufeinander auf. Wenn Sie wenig Qt-Erfahrung haben, sollten Sie den ersten Teil zumindest überfliegen. Der zweite Teil ist eigenständig: Wer sich beispielsweise nur für MeeGo interessiert, kann den Symbian-Teil ohne Konsequenzen überspringen.

Allgemein sei angemerkt, dass das Auswendiglernen von Codekonstrukten und Idiomen in der produktiven Wirtschaft (außer bei der Examensvorbereitung) sinnfrei und unüblich ist.

Gelbe Notizzettelchen kosten im Hunderterpack Centimos und dieses Buch passt auch im engsten Flieger ins Handgepäck – warum also Klassenhierarchien auswendig lernen?

1.4 Wie funktioniert Qt?

An sich ist die Idee von plattformunabhängiger Software nicht neu: Interpretierte Sprachen wie Java werden seit Jahren mit dem Slogan „Write once, run anywhere“ beworben. Sie erreichen dies durch das Zwischenschalten einer virtuellen Maschine, die den „Binärcode“ zur Laufzeit analysiert und ausführt:

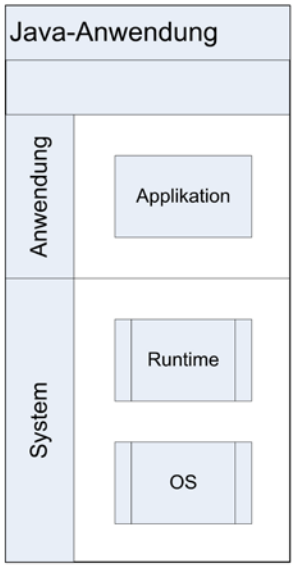


Abbildung 1.1: Die zur Ausführung von Java-Anwendungen erforderliche Runtime ist als separate Anwendung Teil des Betriebssystems

Aus dieser Architektur entstehen leider diverse Nachteile. Erstens kann der Binärcode – ausreichend Zeit und Experimentierfreude vorausgesetzt – in der Regel wieder zu menschenlesbarem Code dekompiliert werden. Außerdem erfordert die Interpretation Rechenleistung: Ein in einer Runtime laufendes Programm ist wesentlich langsamer als ein natives.

Qt geht deshalb einen anderen Weg. Die „Runtime" ist Teil des Programms und mit diesem während der Kompilierung zu einer Binärdatei mit Bibliotheken vereint.

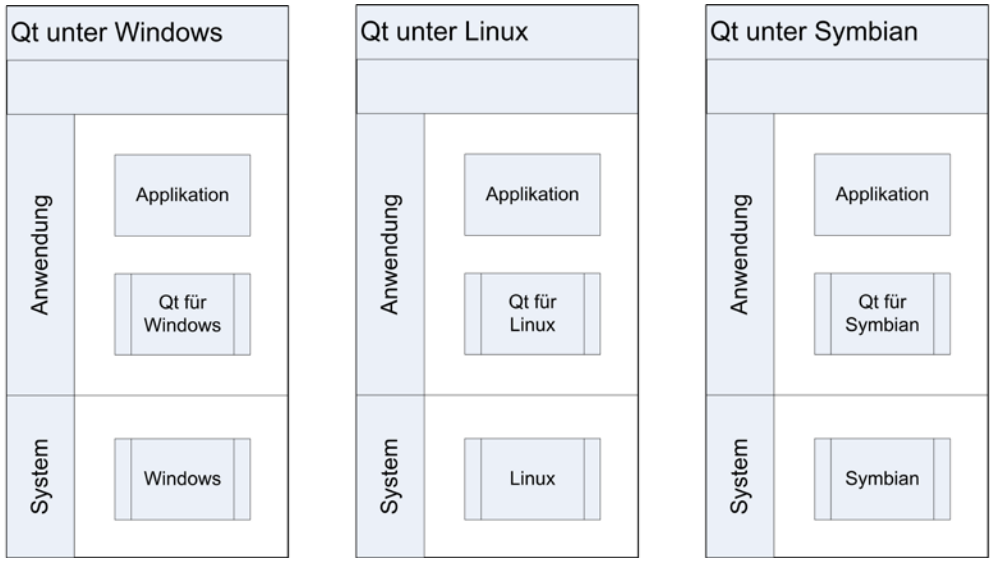


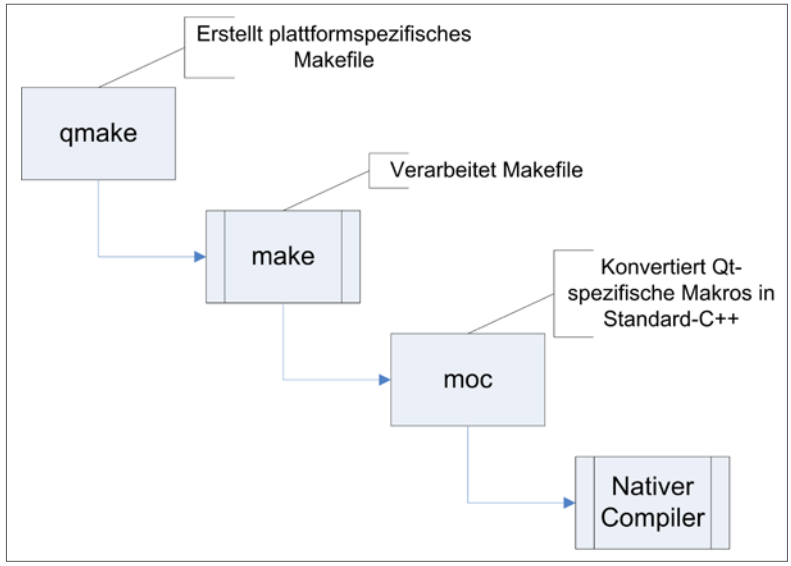
Abbildung 1.2: Qt-Anwendungen enthalten die Runtime als Binärcode

Zur Laufzeit ist ein Qt-Programm nicht mehr von einer nativen Anwendung zu unterscheiden. Als Beispiel hier ein Quellcodeabschnitt aus einem Symbian-Programm, welches sowohl Qt-Befehle als auch Symbian-native APIs aufruft:

```
QtBallZ::QtBallZ(QWidget *parent)
    : QMainWindow(parent)
{
  ui.setupUi(this);

    // lock orientation - S60 code
    CAknAppUi* appUi = dynamic_cast<CAknAppUi*>(CEikonEnv::
                                                    Static()->AppUi());
    if(appUi){
    QT_TRAP_THROWING(appUi ->SetOrientationL( CAknAppUi::
                                            EAppUiOrientationLandscape));
    }

// some more Qt code

    ApplicationPrefs::initPrefs();
    this->showFullScreen();

  myTimerID=startTimer(75);
}
```

Sofern ein Programm keinen plattformspezifischen Code enthält, kann es von Plattform zu Plattform unverändert kompiliert werden – es ist ja nur von der Qt-Bibliothek abhängig.

Die eigentliche Kompilierung erfolgt in mehreren Schritten, die im Diagramm unten zusammengefasst werden:



Im ersten Schritt erstellt qmake aus der Qt-Projektdatei ein plattformspezifisches Makefile. Dieses wird von der plattformeigenen make-Software interpretiert. Bevor jedoch der C++-Compiler auf das Projekt losgelassen wird, wird moc ausgeführt.

Dieser sogenannte Meta Object Compiler analysiert das Qt-Projekt und verwandelt nichtstandardisierte Dateien und Konstrukte in normales C++. Dieses wird danach vom plattformeigenen C-Compiler in eine Plattformanwendung kompiliert, die von einer normalen (nativen) Anwendung nicht unterschieden werden kann.

1.5 Historisches

Wir werden uns kurz mit der Geschichte von Qt beschäftigen. Dies mag auf den ersten Blick langweilig erscheinen, ist aber sinnvoll – es erklärt nämlich die hohe Updatefrequenz.

1.5.1 Version 1

Die Arbeit an Version 1.0 begann im Jahre 1991 – der Name wurde aus ästhetischen Gründen gewählt. 1994 gründeten Haavard Nord und Eirik Chambe-Eng das Unternehmen Trolltech, um Qt zu vermarkten.

Schon damals gab es die duale Lizenzierungsstrategie. Kommerzielle Entwickler kauften eine Lizenz, während nichtkommerzielle Entwickler unter der Free-Qt-Lizenz arbeiteten.

1.5.2 Version 2

Da Free Qt nicht den Standards der FSF entsprach, wurde Qt 2.0 unter einer QPL genannten Lizenz veröffentlicht, die der FSF nach wie vor nicht entsprach. Da der KDE-Desktop zum damaligen Zeitpunkt immer populärer wurde, einigte man sich auf die Gründung der Free Qt Foundation.

Diese Organisation hatte die Aufgabe, die Verfügbarkeit einer quelloffenen Qt-Version „auf immerdar“ sicherzustellen. Zum Erreichen dieses Ziels wurde ihr die Lizenz erteilt, Qt unter einer BSD-Lizenz zu vertreiben, sollte Trolltech für mehr als ein Jahr keine neue quelloffene Version anbieten.

Um diesen „totalen Rechteverlust“ zu vermeiden, bringen Trolltech bzw. Nokia seither im Sechsmonatstakt neue Versionen heraus.

1.5.3 Version 3

Qt 3 war ein größtenteils zu Qt 2.x abwärtskompatibles Upgrade. Es brachte eine neue Datenbankengine sowie ein Übersetzungstool namens Linguist, welches im Buch später genauer behandelt wird.

1.5.4 Version 4.0

Die vierte Version des Frameworks brachte signifikante Änderungen, die teilweise massive Implikationen auf die Abwärtskompatibiltät hatten. Teile von KDE mussten neu geschrieben werden – da die freie Lizenz auf GPL umgestellt wurde, wurden diese Änderungen von der Entwicklergemeinde akzeptiert.

Aus technischer Sicht wies die neue Version des Frameworks diverse Verbesserungen im Bereich des GUI-Designs auf.

1.5.5 Version 4.1

Version 4.1 brachte erstmalig eine betriebssystemunabhängige SVG-Unterstützungsbibliothek mit.

1.5.6 Version 4.2, 4.3

4.2 und 4.3 beinhalteten diverse neue GUI-Widgets.

1.5.7 Version 4.4

In dieser Version von Qt wurde erstmalig WebKit integriert. Dies bringt den Vorteil, dass Qt-Anwendungen einen „integrierten Browser“ zur Hand haben und nicht mehr von der vom Betriebssystem gestellten Infrastruktur anhängig sind.

Weiter wurde ein Multimedia-Framework namens Phonon integriert. Es funktioniert am PC prächtig, über die mobilen Implementierungen sei der Mantel des Schweigens gehüllt.

1.5.8 Version 4.5

Nach der Übernahme durch Nokia wurde die offene Lizenz abermals geändert. Statt der GPL bot man Qt nun unter der LGPL an – dies ermöglichte die Entwicklung von kommerziellen Anwendungen auch ohne den Besitz einer kommerziellen Qt-Lizenz.

1.5.9 Version 4.6

Version 4.6 ist die erste vollkommen unter Nokias Management entstandene Qt-Version. Sie brachte diverse Multitouch-relevante APIs mit, die bis heute nur teilweise implementiert wurden. Auch wurde ein Animations-Framework integriert, welches die Spieleentwicklung erleichtern soll.

Von großer Bedeutung ist hier die Unterversion 4.6.3. Obwohl wesentlich unzuverlässiger als 4.6.2, war sie die erste Version, welche für die Verwendung im Ovi Store freigegeben wurde.

1.5.10 Version 4.7

Qt 4.7 kann am besten als „Qt for Dummies“ beschrieben werden. Eine neue API namens Qt Quick erlaubt die Erstellung von Qt-Benutzerschnittstellen unter QML, einer mit JavaScript eng verwandten Sprache.

1.6 Lizenzrechtliches

Auf Kongressen taucht oft die Frage auf, ob sich der Erwerb einer kommerziellen Lizenz von Qt lohnt. Der Autor vertritt die Meinung, dass dies keinen Sinn macht.

Der einzige Vorteil der kommerziellen Lizenz ist, dass man eigene Änderungen am Framework nicht mit der Allgemeinheit teilen muss. Auf den ersten Blick klingt dies gut – wieso soll ich meine eigenen Bugfixes mit meinen Konkurrenten teilen?

Der Nachteil erschließt sich spätestens beim zweiten oder dritten Update. Nokia nimmt auf private Bugfixes und Änderungen keine Rücksicht. Was nicht im offiziellen Source Tree ist, muss also immer wieder eingepflegt werden – nach einiger Zeit nennt man dann ein eigenes Qt-Wartungsteam sein Eigen …

2 Erste Schritte

Nach den eher theoretischen Betrachtungen im ersten Kapitel geht es hier voll zur Sache. Am Ende dieses Kapitels haben Sie Ihre erste Qt-Anwendung fertig und das Signal-Slot-System verstanden.

2.1 Qt Creator installieren

An sich kann Qt mit fast jeder IDE kombiniert werden. Eclipse, Visual Studio oder was auch immer: Die Wahrscheinlichkeit, dass Trolltech dafür ein Plug-in anbietet, ist hoch. Trotzdem bietet das Unternehmen auch eine eigene Entwicklungsumgebung namens Qt Creator an.

Da davon auszugehen ist, dass neue Features der Plattform erst in Qt Creator integriert werden, lohnt es sich, auf diese Entwicklungsumgebung umzusteigen. Erfreulicherweise ist Qt Creator kostenlos und kann unter *http://qt.nokia.com/downloads/* –> Qt SDK for Windows heruntergeladen werden. Traditionell installiert das SDK gleich die Mobiltoolchains – wer das nicht will, kann sie manuell deselektieren.

Nach der Installation des Qt SDK finden sich diverse neue Programme im Startmenü. Wichtig sind allerdings nur Qt Assistant, Qt Creator, Qt Linguist und die Qt Command Line – betrachten wir sie nacheinander.

2.1.1 Qt Assistant

Qt Assistant ist die Online-Hilfe der Qt-Plattform. Einmal gestartet, präsentiert das Programm eine Liste von Themen und lässt sich wie jedes andere Online-Hilfesystem bedienen.

Wenn man eine Funktion sucht, so gibt man ihren Namen ohne Klammern in das Suchfeld ein und klickt ihren Eintrag an. Das Programm fragt danach wie im Dialog unten gezeigt, um welche Klasse es geht:

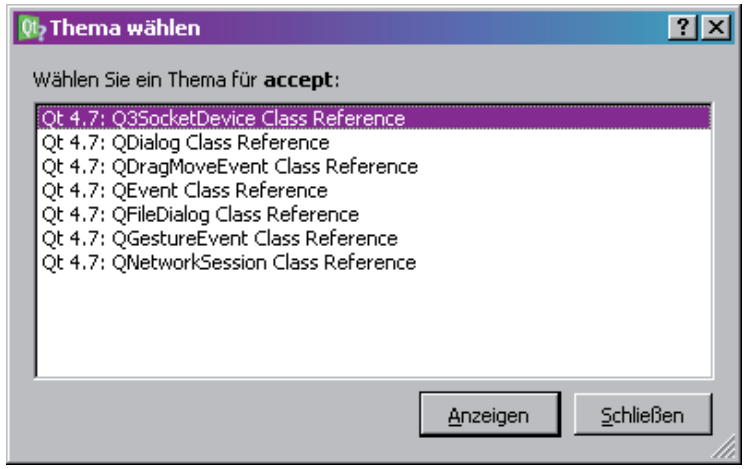


Abbildung 2.1: Die Funktion accept kommt in mehr als einer Klasse vor

2.1.2 Qt Creator

Qt Creator stellt die eigentliche Entwicklungsumgebung dar, in der Qt-Programme entworfen, kodiert und getestet werden.

2.1.3 Qt Linguist

Qt Linguist dient zur Übersetzung von Programmen. Ihm ist ein eigenes Kapitel in diesem Buch gewidmet.

2.1.4 Qt Command Line

Abschließend sei noch kurz auf die Qt Command Line hingewiesen:

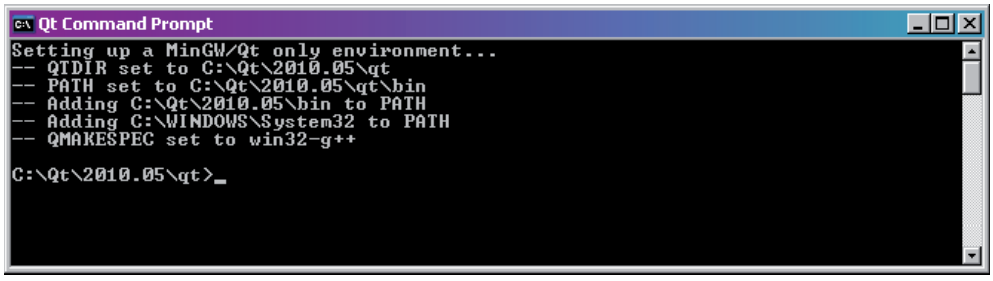


Abbildung 2.2: Im Qt Command Prompt sind die PATH-Variablen der Qt-Installation voreingestellt

Dabei handelt es sich um ein spezielles Command-Fenster, in dem Qt-Befehle direkt aufgerufen werden können. Dies ist insofern sinnvoll, als Qt sich selbst nicht in die PATH-Variable des Systems einträgt …

2.2 Ein neues Projekt

Nun aber genug drumherum geredet – es ist an der Zeit, unser erstes Programm zu entwerfen. Starten Sie Qt Creator – die IDE wird wie im Bild unten gezeigt aussehen:

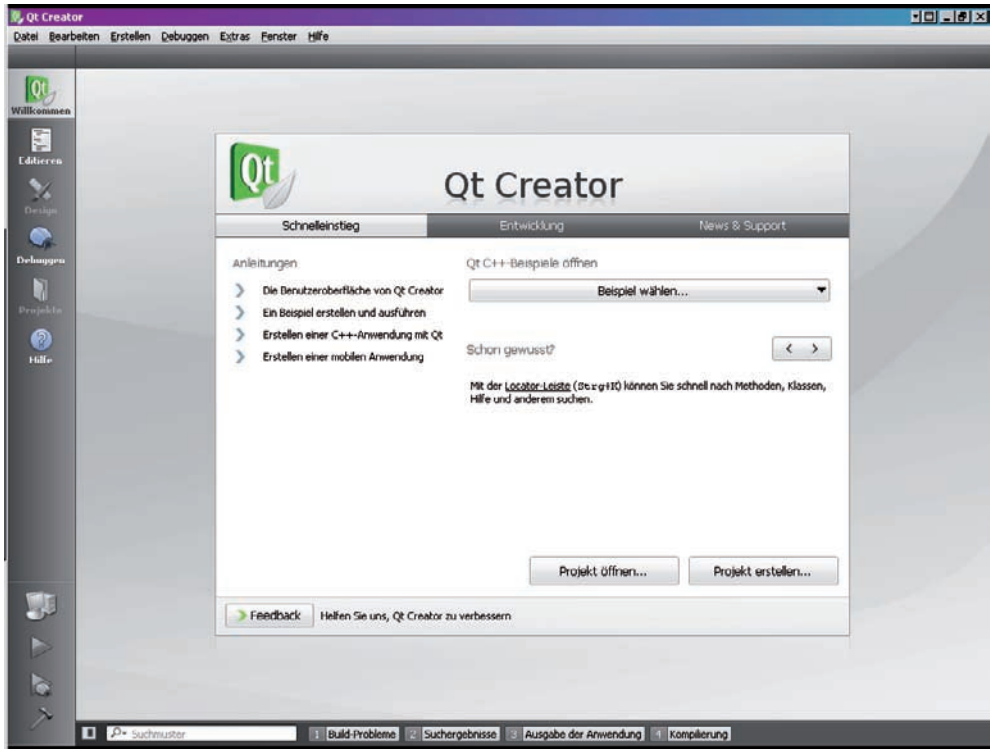


Abbildung 2.3: Qt Creator steht bereit, um Sie bei Ihren programmatischen Verbrechen zu unterstützen

Klicken Sie im Start-Wizard auf Projekt erstellen, um ein neues Projekt zusammenzubauen. Alternativ können Sie die Projekterstellung auch aus dem Menü anstoßen.

Im daraufhin erscheinenden Wizard wählen Sie Qt-GUI-Anwendung.