# 1 Anleitung zur Installation der Qt-Beispiele für Computergraphik bei Prof. Zachmann

Die Anleitung bezieht sich auf die Installation von Qt und der Einbindung der Beispiele unter Visual Studio .NET 2003.

# 2 Vorraussetzungen

Visual Studio .NET 2003 und Qt Version (4.3.2) von Trolltech. Es sollte zuerst Visual Studio installiert werden. Für die Installation / Kompilieren von Qt werden ca. 2 GB freier Platz auf C benötigt.

# 3 Qt

## 3.1 Download

Das Qt Framework kann von http://www.trolltech.com/developer/downloads/qt/windows als Quelldaten, die benötigt werden, geladen werden. Es muss die reine Quellcode Version verwendet werden ohne MinGW. Es empfiehlt sich die Quellen nach C:\Programme\Qt\«Version»\ zu entpacken<sup>1</sup>

## 3.2 Patch von QtWin

Der inoffizielle Patch des QtWin Projektes darf ab der Qt Version 4.3.2 nicht mehr verwendet werden.

#### 3.3 System anpassen

Qt benötigt noch verschiedene Umgebungsvariablen<sup>2</sup>, die auch als Benutzervariablen angelegt werden können. Diese müssen entweder neu angelegt oder passend ergänzt werden:

PATH	$C: \ \ C: \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ $
INCLUDE	$C:\Programme\Qt\@Version\)include$
LIB	$C:\Programme\Qt\@Version\>\lib$
QMAKESPEC	win32-msvc.net <sup>3</sup> $oder$ win32-msvc2005 <sup>4</sup>
QTDIR	$C:\Programme Qt \ll Version \gg$

Um die korrekten Angaben zu prüfen, kann man mit Hilfe des Visual Studio Command Prompt und dem Befehl echo %LIB% (analog die anderen Variablen) die Angaben zu den Libraries sich ausgeben lassen. Sollten hier Probleme auftreten, wird auch das Kompilieren im nächsten Schritt fehlschlagen.

## 3.4 Kompilieren und installieren

Im Visual Studio Command Prompt wechselt man nach C:\Programme\Qt\«Version»\.

- Dort führt man den Befehl *configure* aus. Dieser Vorgang kann je nach Rechner schon einige Zeit in Anspruch nehmen und muss ohne Fehler durchlaufen werden.
- Nach dem Durchlauf kompiliert man nun Q<br/>t mit dem Befehl $\mathit{nmake}.$  Dies kann ebenfalls einige Stunden dauern.
- Um nun erstellte temporäre Dateien zu entfernen, verschiebt man das Verzeichnis C:\Programme\Qt\«Version»\lib aus dem Qt-Ordner, ruft dann im *Command Prompt* den Befehl *nmake clean* auf und verschiebt das *lib*-Verzeichnis wieder zurück.

 $<sup>^1\</sup>mathrm{Der}$ Qt Pfad darf keine Leerzeichen enthalten, da dies zu Problemen führen kann

 $<sup>^2 {\</sup>rm Die}$  Umgebungsvariablen werden unten den Eigenschaften des Arbeitsplatzes, Reiterkarte Erweitert und dem Button Umgebungsvariablen gesetzt

 $<sup>^3{\</sup>rm für}$  Visual Studio . NET 2003

<sup>&</sup>lt;sup>4</sup>für Visual Studio .NET 2005

Danach ist Qt für das System passend kompiliert und installiert

#### 3.5 Erzeugen der Qt und Visual Studio Projektdateien

Man erstellt ein Verzeichnis mit a



3.6 Projekt kompilieren und linken



allen notwendigen Dateien (\*.cpp, \*.h, \*.ui o.ä.). Man wechselt im *Visual Studio Command Prompt* in dieses Verzeichnis und fürt dort folgende zwei Befehle aus:

**qmake -project -t vcapp "Qt += opengl"** hiermit legt man die Qt spezifischen Projektdateien an

**qmake -tp vc** erzeugt die Visual Studio spezifischen Projektdateien

Enthält das Quellcodeverzeichnis Unterverzeichnisse, öffnet man die pro-Datei mit einem Texteditor und trägt unter *INCLUDEPATH* die Unterverzeichnisse ein

Nachdem man nun die Visual Studio Datei geöffnet hat, kann man die Quellcodes editieren. Die Ausführung von *qmake* importiert in das Visual Studio Projekt nur die Standard Headerdateien und Libraries. Für OpenGL-Projekte müssen noch verschiedene Pfade und Libraries ergänzt werden.

#### 3.6.1 Einbinden Headerdateien in das Projekt



Für die Bearbeitung von OpenGL Programmen müssen zusätzlich die QtOpenGL-Headerdateien mit übernommen werden, da sonst Kompilierfehlermeldungen entstehen. Man öffnet dazu die Projekteigenschaften und trägt die fehlenden Include-Verzeichnisse nach, im QtOpenGL-Fall: C:\Programme\Qt\«Version»\include\QtOpenGL. Man kann über die Konfiguration für alle oder einzeln für den Debug und Release die Include-Pfade setzen. Das Beispiel bezieht sich nur auf den Debug Stand, ist aber für die Anwendung völlig ausreichend.

Über den erweiterten Dialog kann man die Verzeichnisse auswählen oder manuell eingeben. Natürlich müssen alle Include-Verzeichnisse angegeben werden. Es gehören auch die Include-Verzeichnisse dazu, die vom Projekt selbst verwendet werden, d.h. es müssen hier auch die Verzeichnisse angegeben werden, die man zuvor in die Qt-Projektdatei eingegeben hat. Absolute oder relative Pfadangaben können verwendet werden.

Damit können nun die Quellcodes kompiliert werden.

#### 3.6.2 Einbinden der Libraries in das Projekt

Configurationalizeration     Adaman     Standardbildenkein spronteren	Konfiguration: Alle Konfigura	ationen 💌 Plattform: Active(Win32)	•	Konfigurations-Manager
Algemein Debugen C (C++ Algemein Dubugen Dubugen Debug	Konfigurationseigenscha	Zusätzliche Abhängigkeiten	c:\Programme\Qt\4.3	3.2\lib\gtmaind.lib c:\Program
Debuggen C/C++     Boliothek ignoreren       Algemein Eingabe Debuggen System Optimiung Eingabettale EIN.     Weizdert Ressourchabet inbetten Symbolerrweise erzwingen Verzagert geladene DLLs       wizitzliche Abhängigkeiten Eingabettale EIN.     EXECUTION Verzagert geladene DLLs wizitzliche Abhängigkeiten Eingannen (Qf4-3.2.Vib/QCopenGL4.8b Eingannen (Qf4-3.2.Vib/QCopenGL4.8b Eingannen (Qf4-3.2.Vib/QCopenGL4.8b Eingannen (Qf4-3.2.Vib/QCopenGL4.8b Eingannen (Microsoft Piotom SDK-7LB/QDepGL4.2.1b)         Eingannen (Microsoft Piotom SDK-7LB/QDepGL4	Allgemein	Standardbibliotheken ignorieren	Nein	
Modudéminon     Modudémin	Debuggen	Bibliothek ignorieren		
Gardiner       Aligemein         Aligemein       Woduł zur Assembly hinzufügen         Symbolwerweise erzwingen       Symbolwerweise erzwingen         Debuggen       Symbolwerweise erzwingen         Optimierung       Verzögert geladene DLLs         Signatur       Verzögert geladene DLLs         VirogenmelQt(4.3.2.Vib/Qtcuid4.lib       Verzögert geladene DLLs         VirogernmelQt(4.3.2.Vib/Qtcuid4.lib       Verzögert geladene DLLs         Verzögert geladene DLLs       Verzögert geladene DLLs         Verzögert geladene DLLs       Verzögert geladene DLLs         Verzöger	C/C++	Moduldefinition		
Algemein Engaber Pebuggen System Orpogrammel (VM-3.2)Lib) (QrbenGL32.Lib) Expropriet for SDK-YLib) (OpenGL32.Lib) Exprogrammel (VM-3.2)Lib) (QrbenGL32.Lib) Exprogrammel (VM-3.2)Lib) (QrbenGL	🔄 Linker	Modul zur Assembly hinzufügen		
Eingabe Debuggen System Optimieung Eingabeteites III. Syroogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib ://roogramme(Qf4.3.2.Uib/Qcored.lib)	Allgemein	Verwaltete Ressourcendatei einbetten		
Debuggen System Optimierung Engebertete ZDL     Verzägert geladene DLLs       usätzliche Abhängigkeiten     Image: Compare (Qrd, 3.2)kil/Qumaind.lib c:/Programme (Qrd, 3.2)kil/Qumaind.lib c:/Programme (Qrd, 3.2)kil/Qumaind.lib c:/Programme (VM:crosoft Platform SDK?\Lib/OpenGL32.lib c:/Programme (Microsoft Platform SDK?\Lib/OpenGL32.lib	Eingabe	Symbolverweise erzwingen		
System Optimiurug Engeberden DZL usätztiche Abhängigkeiten EVPorgannme(Qt4.3.2)(b)(Qtcu4.1b) EVPorgannme(Qt4.3.2)(b)(Qtcu64.1b) EVPo	Debuggen	Verzögert geladene DLLs		
usätzliche Abhängigkeiten ExPorgammelQV4.3.2libiQuend4.lib ExPorgammelQV4.3.2libiQUecored4.lib ExPorgammelQV4.3.2libiQUecored4.lib ExPorgammelQV4.3.2libiQUecored4.lib ExPorgammelQV4.3.2libiQUecored4.lib ExporgammelVMcrosoft Platform SDK/VLibiQDenGC32.libi	Ontimierung			
usätztiche Abhängigkeiten  CVPogramme(Qf(4.3.2)(ib)(dfmaind.lib ://Pogramme(Qf(4.3.2)(ib)(dfcaid4.lib ://Pogramme(Qf(4.3.2)(ib)(Cored4.lib) ://Pogramme(Qf(4.3.2)(ib)(Cored4.lib) ://Pogramme(Qf(4.3.2)(ib)(OpenGL32.lib) ://Pogramme(Qf(4.3.2)(ib)(OpenGL32.lib)) ://Pogramme(Qf(4.3.2)(ib)(OpenGL32.lib))) ://Pogramme(Qf(4.3.2)(ib)(OpenGL32.lib))) ://Pogramme(Qf(4.3.2)(ib)(OpenGL32.lib))) ://Pogramme(Qf(4.3.2)(ib)(OpenGL32.lib))) ://Pogramme(Qf(4.3.2)(ib)(Ib)(Ib)(Ib)(Ib)(Ib)(Ib)(Ib)(Ib)(Ib)(I	Eingebettete IDL			
usätztiche Abhängigkeiten  C:\Programme\(?(4.3.2)lib)(dimaind.lib c:\Programme\(?(4.3.2)lib)(Cdcu64.lib c:\Programme(?(4.3.2)lib)(Cdcu64.lib c:\Programme(?(4.3.2)lib)(Cdcu64.lib c:\Programme(?(4.3.2)lib)(Cdcu64.lib c:\Programme(?(4.3.2)lib)(Cdcu64.lib c:\Programme(?(4.3.2)lib)(Cdcu64.lib c:\Programme(?(4.3.2)lib)(Cdcu64.lib c:\Programme(?(4.3.2)lib)(Cdcu64.lib c) ererbe Wette:				
C:\Programme\(Qt(4.3.2)(b)(qtmaind.lib C:\Programme\(Qt(4.3.2)(b)(Qtc)(dt,lib C:\Programme\(Qt(4.3.2)(b)(Qtc)(dt,lib C:\Programme\(Qt(4.3.2)(b)(Qtc)(dt,lib)(D	usätzliche Abhängig	keiten		
://Yorginme/Qt4.3.2lbiQtCode4.lb ://Yorginmi/Qt4.3.2lbiQtCode4.lb ://Yorginmi/Qt4.3.2lbiQtCode4.lb ://Yorginmi/Qt4.3.2lbiQtCode6.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQtOpenGL4.lb ://Yorginmi/Qt4.3.2lbiQt0penGL4.lb ://Yorginmi/Qt4.lb ://Yorginmi/Qt4.lb ://Yorginmi/Qt4.lb ://Yorginmi/Qt4.lb ://Yorginmi/Qt4.lb	::\Programme\Qt\4.3.2\lib	∖qtmaind.lib		
c:/Programme/Qt4.3.2/bi/QtCored4.lb E:/Programme/Qt4.3.2/bi/QtCored4.lb c:/Programme/Microsoft Platform SDK7/Lb/OpenGL32.lb	c:\Programme\Qt\4.3.2\lib	QtGuid4.lib		
C Standard Control (2014) 22 (2010) (2010) (2014	c:\Programme\Qt\4.3.2\lib	\QtCored4.lib		
C:\Programme\'Microsoft Platform SDK"\Lib\OpenGL32.lib	c:\Programme\Qt\4.3.2\lib	\QtOpenGL4.lib		
< <p>Provide the second se</p>	c:\Programme\"Microsoft	Platform SDK"\Lib\OpenGL32.lib		
Cereta Verte:				
C P				
Cereta Werte:				
C C				
ererbte Werte:			~	
	<		~	
cernel s2 lin	<pre>ererbte Werte:</pre>		×	

Damit nun keine Linkerfehler auftreten muss das Projekt noch gegen die passenden Windowslibraries gelinkt werden. Für die OpenGL Anwendung einmal gegen die die QtOpenGL Library und nachfolgend gegen die windowseigene OpenGL32 Library. Im Dialog der Projekteigenschaft wird dazu dann unter dem Menüpunkt *Linker*, dort der Menüpunkt *Eingabe* und das Feld *Zusätzliche Abhängigkeiten* die passenden Dateien hinzugefügt. Für OpenGL sind folgende Dateien zu wählen:

- C:\Programme\Qt\«Version»\lib\QtOpenGLd4.lib
- C:\Programme\"Microsoft Platform SDK" \Lib\OpenGL32.lib

Bei den Qt-Libraries ist auf das d im Dateinamen zu achten, entweder müssen alle Dateinamen ohne d oder mit d angegeben werden. Unterschiedliche Angaben führen dazu, dass zwar ein Executable erzeugt wird, dieses aber nicht startet. Der *qmake* Prozess fügt automatisch Dateien mit d hinzu. Natürlich können auch hier wieder die verschiedenen Konfigurationen verwendet werden. Danach ist die Konfiguration abgeschlossen und man kann das Projekt nun vollständig kompilieren.