Kapitel 3

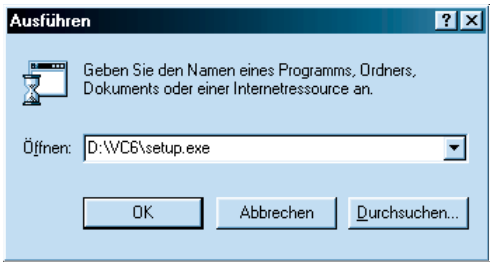
Wie erstellt man eigene Programme?

In diesem Kapitel geht es darum, dass Sie Ihren Compiler und Ihre Programmierumgebung kennen lernen. Stellvertretend für viele andere leistungsfähige Compiler mit integrierter Entwicklungsumgebung stelle ich Ihnen hierzu den Visual C++-Compiler vor, den Sie von der Buch-CD aus installieren können. Leider kann der Visual C++-Compiler nur unter Windows installiert werden. Linux-Anwender müssen deshalb aber nicht verzweifeln: höchstwahrscheinlich ist auf ihrem System bereits der GNU-C++-Compiler installiert, zu dem es am Ende dieses Kapitels eine kleine Einführung gibt. Lesen Sie dieses Kapitel aufmerksam durch, denn die beschriebenen Techniken brauchen Sie zur Nachprogrammierung der in den folgenden Kapiteln vorgestellten Programme.

Installation des Visual C++-Compilers

Bevor Sie das Setup-Programm der Autoren-Edition von Visual C++ aufrufen, sollten Sie alle anderen Programme schließen. Speichern Sie vor allem Ihre Daten und Dateien, da Windows im Laufe der Installation neu gestartet wird.

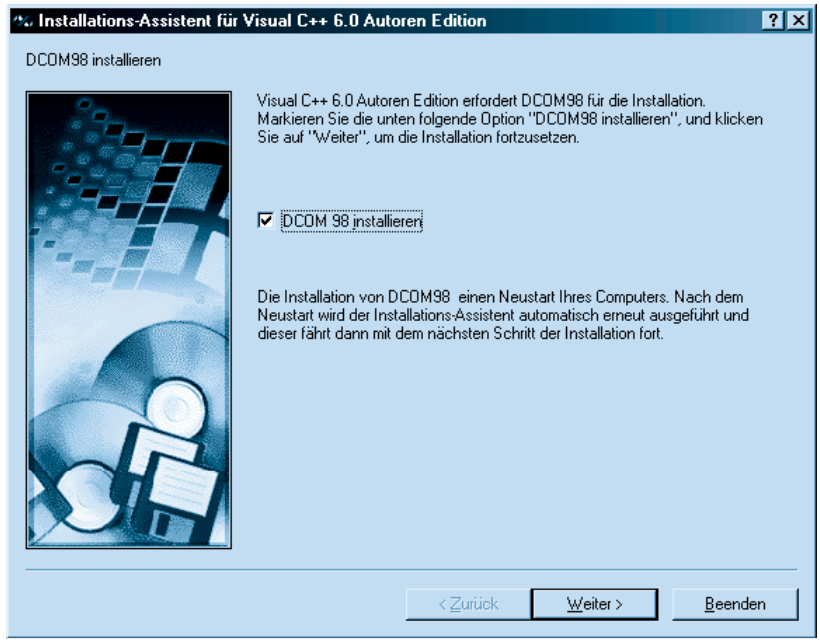
1 Legen Sie die Buch-CD in Ihr CD-ROM-Laufwerk ein.



> **Hinweis**
>
> *Das Setup-Programm befindet sich im Unterverzeichnis* VC6 *der Buch-CD. Statt* D:\ *müssen Sie den Laufwerksbuchstaben Ihres CD-ROM-Laufwerks eingeben.*

2 Öffnen Sie das START-Menü von Windows und wählen Sie den Befehl AUSFÜHREN. Im gleichnamigen Dialogfenster tippen Sie den Pfad zum Setup-Programm von Visual C++ ein oder wählen die *setup.exe*-Datei über den Schalter DURCHSUCHEN und den zugehörigen Suchdialog aus. Schicken Sie den Dialog mit einem Klick auf den OK-Schalter ab.

Es erscheinen jetzt nacheinander das Begrüßungsfenster des VC-Installations-Assistenten, der Lizenzvertrag und die Abfrage der Benutzerdaten. Klicken Sie jeweils auf WEITER, um mit der Installation fortzufahren.



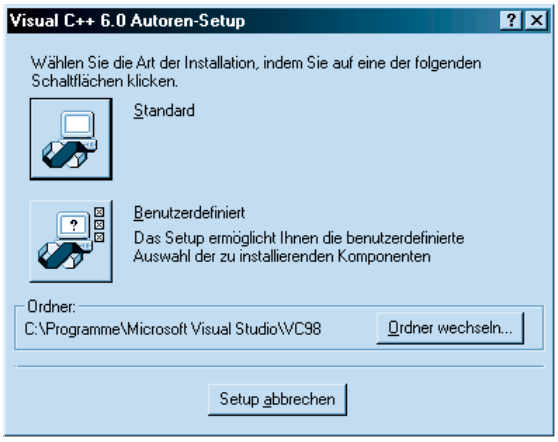
Was ist das?

DCOM *ist eine Windows-Technologie zur Unterstützung verteilter Anwendungen. Die Erstellung verteilter Anwendungen gehört zu den höheren Weihen der Programmierkunst und spielt für die Programme, die wir im Rahmen dieses Kurses erstellen wollen, keine Rolle. Sie müssen DCOM aber zur Unterstützung der VC-Installation installieren lassen.*

3 Falls der Installations-Assistent feststellt, dass auf Ihrem System keine DCOM-Unterstützung eingerichtet ist, fordert er Sie jetzt auf, dies nachzuholen. Achten Sie darauf, dass die Option DCOM 98 INSTALLIEREN markiert ist, klicken Sie dann auf WEITER und lassen Sie Ihren Rechner vom Installations-Assistenten neu starten.

4 Nach dem Neustart werden Sie aufgefordert, einen Ordner für gemeinsam verwendete Dateien anzugeben. Meist empfiehlt es sich, die Vorgabe zu übernehmen.

Jetzt wird der Installations-Assistent beendet und das eigentliche Setup-Programm wird gestartet. Nach einigen einführenden Dialogfenstern werden Sie aufgefordert, die Installationsart festzulegen.

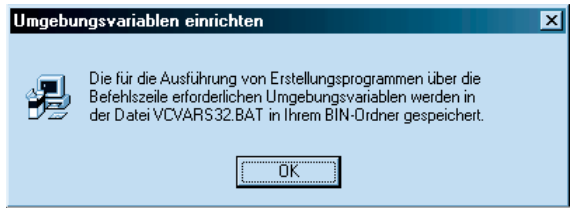


5 Legen Sie zuerst das Installationsverzeichnis für den Visual C++-Compiler fest. Am einfachsten ist es, den vorgegebenen Ordner beizubehalten. Wenn Sie ein anderes Verzeichnis wünschen, klicken Sie auf den Schalter ORDNER WECHSELN.

Als Nächstes legen Sie den Installationsumfang fest. Für unsere Belange empfiehlt sich die STANDARD-Installation.

> **Tipp**
>
> *Wenn Sie mit Festplattenspeicher geizen müssen, klicken Sie auf den Schalter für die BENUTZERDEFINIERTE Installation und deaktivieren Sie die Optionen ACTIVEX-STEUERELEMENTE, DIENSTPROGRAMME, DATENZUGRIFF und eventuell auch MFC UND VORLAGEBIBLIOTHEKEN FÜR VC++. Das sollte Ihnen ungefähr 70 MByte Festplattenspeicher sparen.*

Das Setup-Programm beginnt nun mit dem Kopieren der Dateien. Ist dieser Vorgang abgeschlossen, erscheint noch einmal ein Dialogfeld.



> **Was ist das?**
>
> *Wenn Sie den VC-Compiler von der Konsole (d.h. der MS-DOS-Eingabeaufforderung) aus bedienen wollen, müssen Sie vorab die Batch-Datei* VCVARS32.BAT *ausführen. Danach können Sie den VC-Compiler,* cl.exe*, von jedem Verzeichnis Ihres Rechners aus aufrufen. Wenn Sie – wie im Folgenden beschrieben – mit der integrierten Entwicklungsumgebung von Visual C++ arbeiten, ist die Datei* VCVARS32.BAT *ohne Belang für Sie.*

6 Klicken Sie auf OK, um die Umgebungsvariablen einrichten zu lassen und die Installation abzuschließen.

7 Starten Sie Windows neu.

8 Installieren Sie nach dem Neustart noch die Autorendokumentation (MSDN), wenn Sie die Online-Hilfe zu Visual C++ nutzen möchten (was zu empfehlen ist).

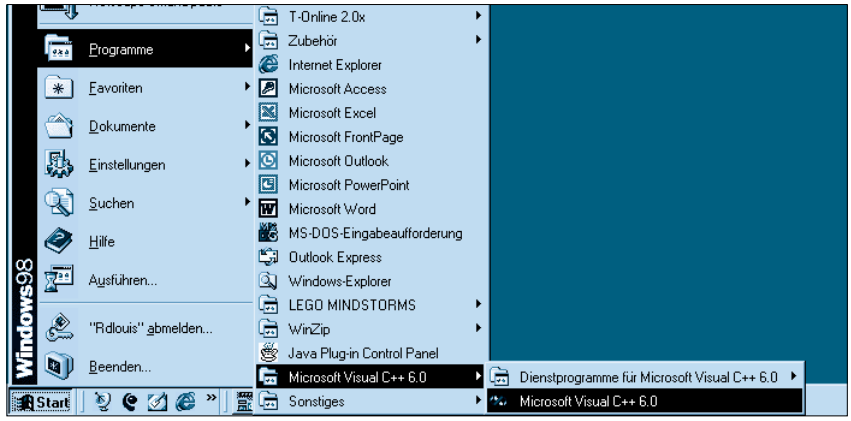
> **Hinweis**
>
> *Bei Bedarf kann die MSDN-Dokumentation auch zu einem späteren Zeitpunkt über das Setup-Programm im MSDN-Unterverzeichnis der CD-ROM installiert werden.*

Visual C++ starten

Visual C++ ist ein Compiler mit einer integrierten Entwicklungsumgebung (kurz IDE für *Integrated Developing Environment*). Für uns als Programmierer bedeutet dies, dass wir auch bei der Programmierung nicht auf den Komfort einer grafischen Benutzeroberfläche verzichten müssen.

> **Was ist das?**
>
> *Bei der Programmerstellung ist der Programmierer auf eine Reihe von Hilfsprogrammen angewiesen (Editor, Compiler, Linker, Debugger), von denen die meisten traditionell Befehlszeilenprogramme sind (Compiler, Linker, Debugger), d.h. sie verfügen über keine Fenster oder grafische Oberflächen und müssen von der Konsole aus (MS-DOS-Eingabeaufforderung) ausgeführt werden (vergleiche GNU-Compiler am Ende dieses Kapitels). Die* integrierte Entwicklungsumgebung *von Visual C++ ist eine Art Über-Programm, von dem aus alle für die Programmentwicklung benötigten Programme aufgerufen werden können. Sie verfügt über einen integrierten Editor, Dialogfenster und Menübefehle zum Aufruf des Compilers und eine ausgereifte Projektverwaltung.*

Starten Sie nun die Visual C++-Entwicklungsumgebung, um sich mit ihr vertraut zu machen.



1 Klicken Sie in der Taskleiste auf die Schaltfläche START und rufen Sie den Eintrag des Visual C++-Compilers aus der Visual C++-Programmgruppe auf.

Nach kurzer Ladezeit erscheint die integrierte Entwicklungsumgebung von Visual C++.

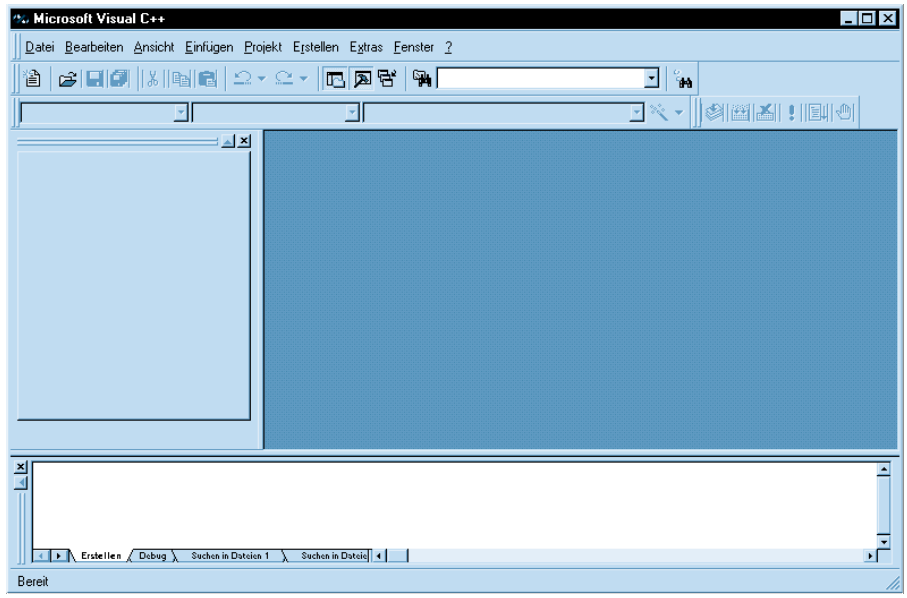


Abbildung 3.1: Die integrierte Entwicklungsumgebung von Visual C++

Neben der Menüleiste, mit den vielversprechend klingenden Menüeinträgen PROJEKT und ERSTELLEN, und einer Reihe von leeren Fensterbereichen gibt es

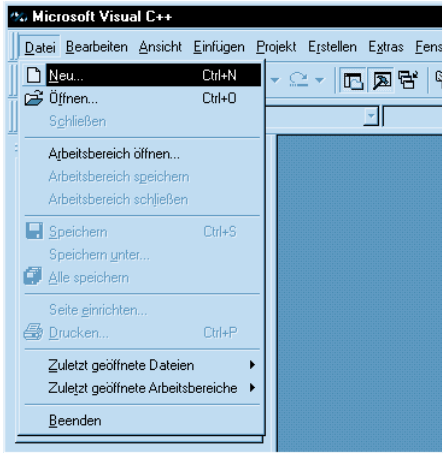
nicht viel zu sehen. Doch das ändert sich, wenn wir mit dem Programmieren beginnen.

Ein neues Projekt anlegen

In Visual C++ werden Programme in Form von Projekten verwaltet (die wiederum in Arbeitsbereichen organisiert werden).

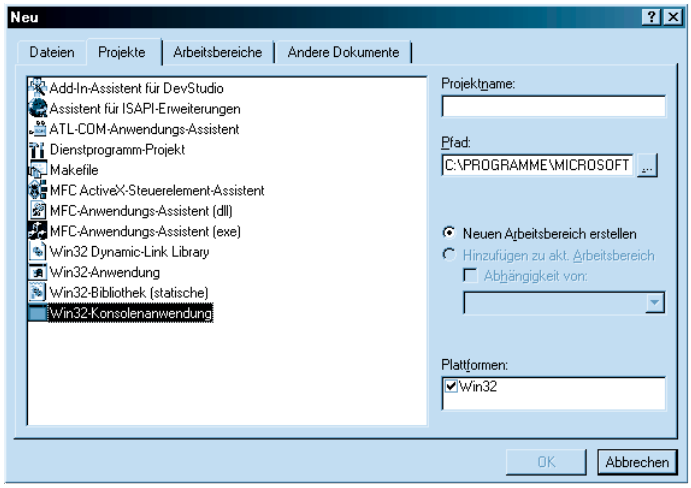
> **Was ist das?**
>
> *Die Quelldateien und Informationen, die zur Erstellung eines Programms benötigt werden, verwaltet der Visual C++-Compiler in Form eines* Projekts. *Die Projektverwaltung ist für den Programmierer umso wertvoller, je komplexer und umfangreicher die erstellten Programme sind.*

Der erste Schritt bei der Programmerstellung mit Visual C++ besteht daher darin, ein passendes Projekt anzulegen.



1 Öffnen Sie das Menü DATEI, klicken Sie auf den Befehl NEU.

> **Tipp**
>
> *Am schnellsten beginnen Sie ein neues Projekt mit der Tastenkombination* Strg N.



2 Im Dialogfeld NEU sollte jetzt die PROJEKTE-Seite angezeigt werden. Falls nicht, klicken Sie einfach auf das gleichnamige Register. Wählen Sie als Projekttyp WIN32-KONSOLENANWENDUNG aus.

Was ist das?

Konsolenanwendungen *sind Programme ohne grafische Oberfläche, die üblicherweise von der Betriebssystemkonsole (unter Windows die MS-DOS-Eingabeaufforderung) aus aufgerufen werden.*

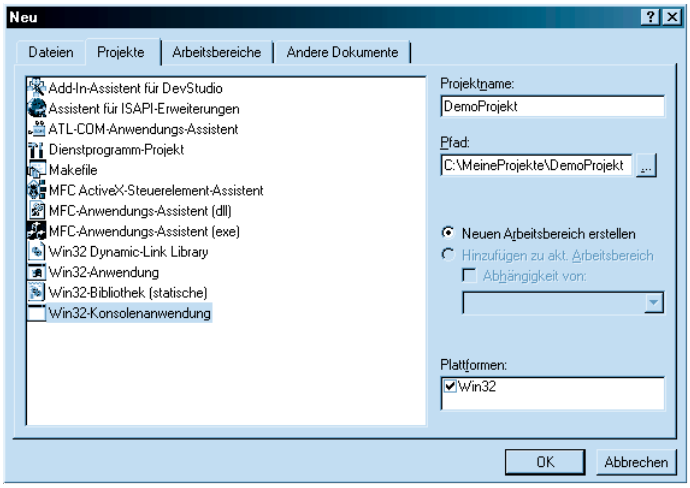
Die Auswahl des Projekttyps ist ein ganz wichtiger Schritt. Durch den Projekttyp teilen wir Visual C++ mit, was wir programmieren wollen: ein eigenständiges Programm, eine Bibliothek, die von anderen Programmen verwendet werden kann, oder vielleicht sogar ein Hilfsprogramm zur Erweiterung der IDE.

Sie sehen: Der Visual C++-Compiler bietet uns weit mehr Möglichkeiten, als wir im Moment nutzen können. Macht nichts, wir konzentrieren uns auf unser eigentliches Ziel: die Erstellung eines Programms. Hierfür stehen drei Projekttypen zur Verfügung:

- MFC-ANWENDUNGS-ASSISTENT (EXE)
- WIN32-ANWENDUNG
- WIN32-KONSOLENANWENDUNG

Die ersten beiden Projekttypen sind für die Erstellung »echter« Windows-Programme gedacht, deren Benutzeroberfläche aus Fenstern aufgebaut ist. Für einen Einführungskurs in C++ ist uns das etwas zu anspruchsvoll, da wir neben dem Einstieg in die Programmiersprache C++ auch noch die Einarbeitung in die Windows-Programmierung und den Aufbau grafischer Benutzeroberflächen mit Fenster, Menüs, Steuerelementen und Mausunterstützung bewältigen müssten.

Bescheiden wir uns also mit der Erstellung von Konsolenanwendungen. Konsolenanwendungen haben keine grafische Benutzeroberfläche, ja im Grunde genommen verfügen sie über gar keine eigene Benutzeroberfläche. Stattdessen nutzen sie zur Ein- und Ausgabe die Konsole, von der sie aufgerufen wurden (unter Windows ist dies die MS-DOS-Eingabeaufforderung). Konsolenanwendungen haben aber den Vorteil, dass man sich ganz auf den reinen C++-Code konzentrieren kann. Im Übrigen werden Konsolenanwendungen auch heute noch eingesetzt – beispielsweise als Hilfsprogramme für Webserver. Und wenn Sie später zur Window-Programmierung überwechseln wollen ... die reine C++-Programmierung bleibt stets die gleiche, Sie müssen nur noch lernen, wie man Programme Window-fähig macht.



3 Geben Sie im Eingabefeld rechts oben einen Namen für Ihr Projekt ein (im Beispiel *DemoProjekt*). Geben Sie darunter an, unter welchem Verzeichnispfad das Projekt gespeichert werden soll. Sie können den Pfad eintippen oder ihn über die Schaltfläche neben dem PFAD-Eingabefeld auswählen. Sie können auch neue Pfade angeben, die Visual C++ für Sie anlegt.

Bestätigen Sie dann mit OK.

Tipp

Erstellen Sie Ihre Projekte unter einem gemeinsamen Pfad – beispielsweise C:\MeineProjekte *– und lassen Sie unter diesem übergeordneten Verzeichnis für jedes Projekt ein eigenes Unterverzeichnis anlegen. So trägt das Projekt aus diesem Kapitel beispielsweise den Namen* DemoProjekt *und die Dateien des Projekts werden in dem Verzeichnis* C:\MeineProjekte\DemoProjekt *abgespeichert.*

Hinweis

Die weiteren Programme aus diesem Buch sind nach Kapiteln geordnet (die Programme jeden Kapitels stehen in einem eigenen Unterverzeichnis).

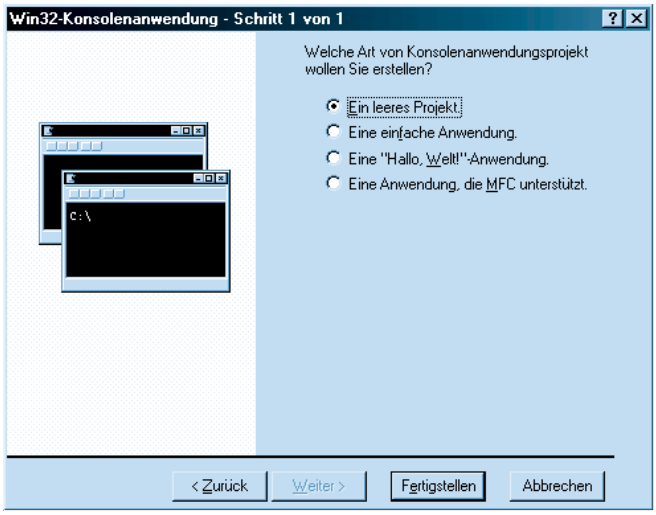
Sicher ist Ihnen die Option NEUEN ARBEITSBEREICH ERSTELLEN aufgefallen. Visual C++ organisiert Projekte in Arbeitsbereichen. Für fortgeschrittene Programmierer ist dies interessant, insofern sie so Ihre Projekte in Gruppen zusammenfassen können – beispielsweise ein Programm und eine für das Programm geschriebene Bibliothek oder die verschiedenen Versionen eines Programms. Für uns sind die Arbeitsbereiche weniger interessant und wir vereinfachen uns die Sache, indem wir bei der Erstellung jedes neuen Projekts die Option NEUEN ARBEITSBEREICH ERSTELLEN markieren, damit Visual C++ die einzelnen Projekte in jeweils eigenen Arbeitsbereichen erstellt.

Hinweis

Wenn Sie trotzdem mit Arbeitsbereichen arbeiten wollen, habe ich zwei Hinweise für Sie: Um ein neues Projekt im Arbeitsbereich eines bestehenden Projekts anzulegen, laden Sie das bestehende Projekt und aktivieren dann im Dialogfeld NEU die Option HINZUFÜGEN ZU AKT. ARBEITSBEREICH.

Über den Menübefehl PROJEKT/AKTIVES PROJEKT FESTLEGEN können Sie auswählen, welches Projekt im Arbeitsbereich Sie bearbeiten wollen.

Nach dem Abschicken des Dialogfelds NEU erscheint ein weiteres Dialogfeld, in dem Sie zwischen verschiedenen Projektgrundgerüsten wählen können.



4 Markieren Sie die Option EIN LEERES PROJEKT und klicken Sie auf FERTIGSTELLEN.

> **Hinweis**
>
> *Die anderen Optionen können Sie nach der Bearbeitung dieses Kapitels gerne einmal selbst ausprobieren. Die IDE erzeugt dann nicht nur ein neues Projekt, sondern legt für Sie auch gleich eine Quelldatei mit einem mehr oder weniger ausführlichen Codegerüst an.*

5 Zum Abschluss fasst Visual C++ noch einmal alle Informationen zu dem Projekt in einem letzten Dialogfeld zusammen. Drücken Sie den OK-Schalter, um das Projekt anlegen zu lassen.

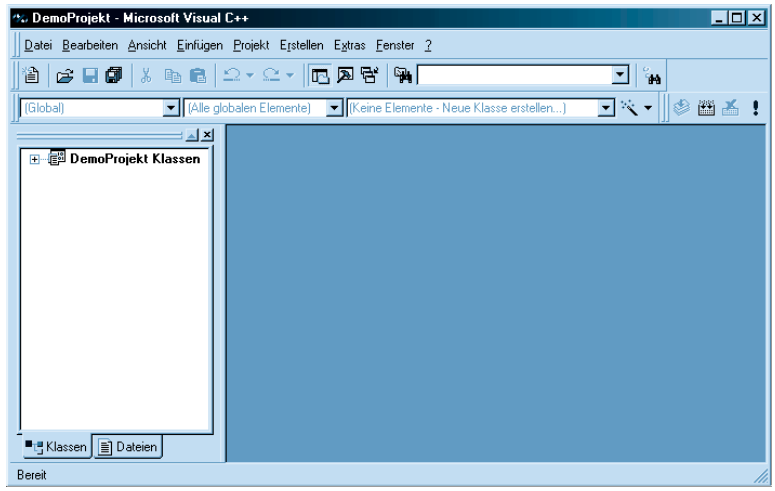


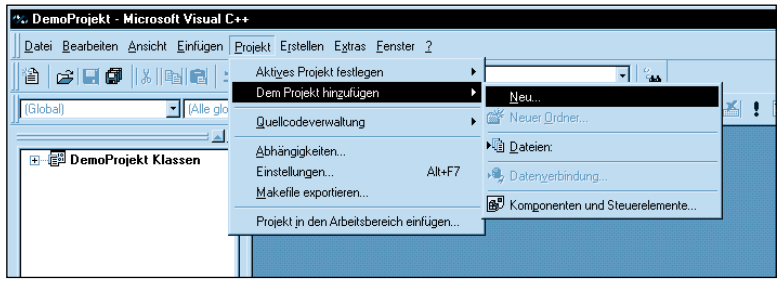
Abbildung 3.2: Die IDE nach dem Anlegen des Projekts

In der IDE hat sich jetzt etwas getan. In der Titelleiste kann man lesen, welches Projekt gerade bearbeitet wird, und im Arbeitsbereichfenster (dies ist das Fenster links unter den Symbolleisten) kann man sich über den Aufbau des Projekts informieren. Im Moment gibt es dort aber noch nicht viel zu sehen, weil unser Projekt noch leer ist.

Den Quelltext aufsetzen

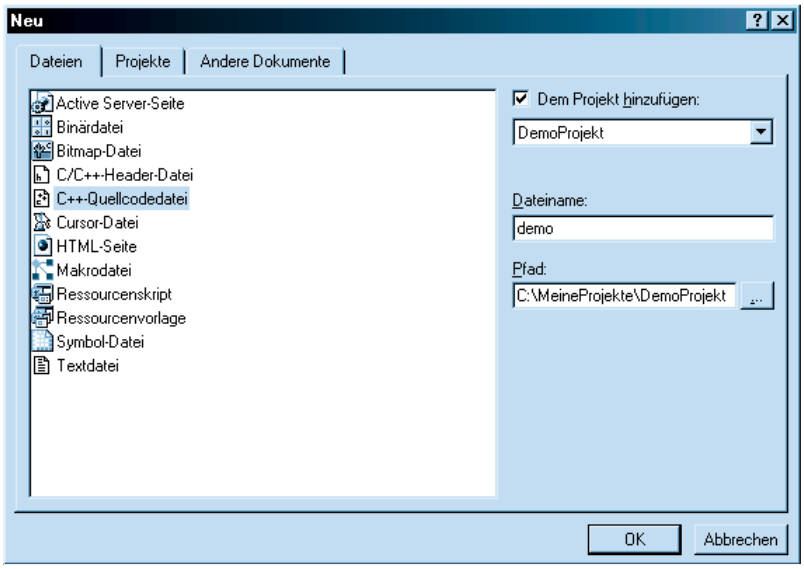
Bevor wir den Quelltext zu unserem Programm eingeben können, müssen wir eine Quelldatei in unser Projekt aufnehmen.

Quelltextdateien anlegen



1 Rufen Sie den Menübefehl PROJEKT/DEM PROJEKT HINZUFÜGEN/NEU auf.

Es erscheint das Dialogfeld NEU mit der Seite DATEIEN.



2 Geben Sie im Feld DATEINAME einen Namen für die hinzuzufügende Datei ein (hier *demo*) und wählen Sie als Dateityp C++-QUELLCODEDATEI aus. Bestätigen Sie mit OK.

Die Quelldatei wird automatisch in den integrierten Editor geladen.

Die Dateien eines Projekts

In linken Teil der Entwicklungsumgebung von Visual C++ wird das Arbeitsbereichfenster angezeigt. (Falls nicht, können Sie es über den Menübefehl ANSICHT/ARBEITSBEREICH aufrufen.) In diesem Fenster können Sie sich darüber informieren, welche Dateien zu Ihrem Projekt gehören.

1 Klicken Sie am unteren Rand des Arbeitsbereichfensters auf das Register DATEIEN.

Das DATEIEN-Register ist – dem Windows Explorer vergleichbar – hierarchisch organisiert. Der oberste Knoten bezeichnet den Arbeitsbereich. Darunter folgen die Knoten der Projekte.

2 Expandieren Sie den Knoten des Projekts, indem Sie auf das Pluszeichen vor dem Knoten klicken.

3 Expandieren Sie den Knoten QUELLCODEDATEIEN.

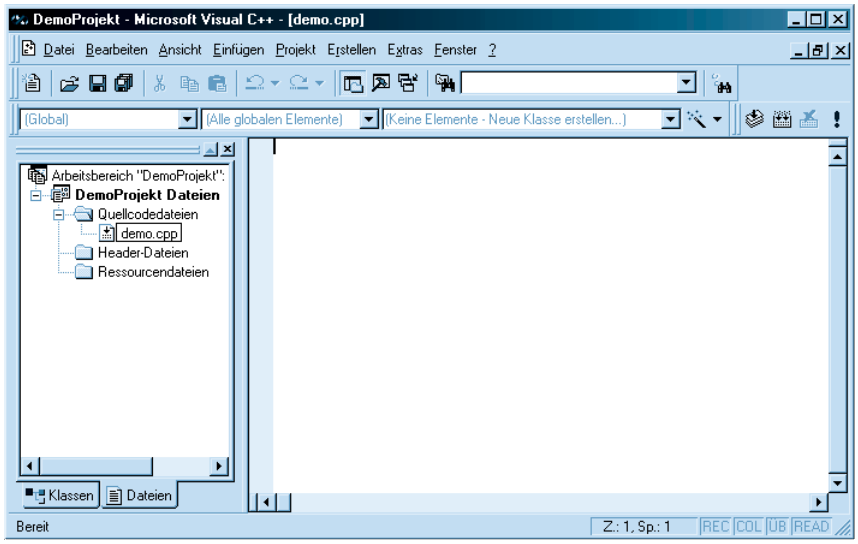


Abbildung 3.3: Arbeitsbereichfenster und geladene Quelldatei in der IDE

> **Tipp**
>
> *Durch Doppelklick auf den Knoten einer Quelldatei können Sie diese zur Bearbeitung in den Editor laden.*

Den Programmquelltext eingeben

Nachdem die Datei als Teil des Projekts angelegt und in den Editor geladen wurde, können Sie den Quelltext des Programms eingeben.

> **Hinweis**
>
> *Wenn Sie nicht sicher sind, ob die richtige Datei in den Editor geladen wurde, schauen Sie einfach in die Titelleiste der Visual C++-IDE. Dort wird der Name der gerade geladenen Datei in Klammern angezeigt. Oder doppelklicken Sie einfach noch einmal im Arbeitsbereichfenster auf den Knoten der Datei.*

1 Tippen Sie den folgenden Programmquelltext ein.

```
// Hallo Welt-Programm

#include <iostream>
using namespace std;

int main() {

   cout << "Hallo Welt!" << endl;

   return 0;
}
```

Tippen Sie den Quelltext bitte genauso ab, wie er oben aufgelistet ist. Was dieser Code im Einzelnen bedeutet, werden Sie im nächsten Kapitel erfahren. Wenn Sie unsicher sind, ob Sie den Quelltext richtig abgetippt haben, kopieren Sie den Quelltext einfach aus der Datei *demo.cpp* auf der Buch-CD.

> **Achtung**
>
> *In C++ wird zwischen Groß- und Kleinschreibung unterschieden. Wenn Sie also beispielsweise* `Main()` *statt* `main()` *eintippen, ist dies ein Fehler!*

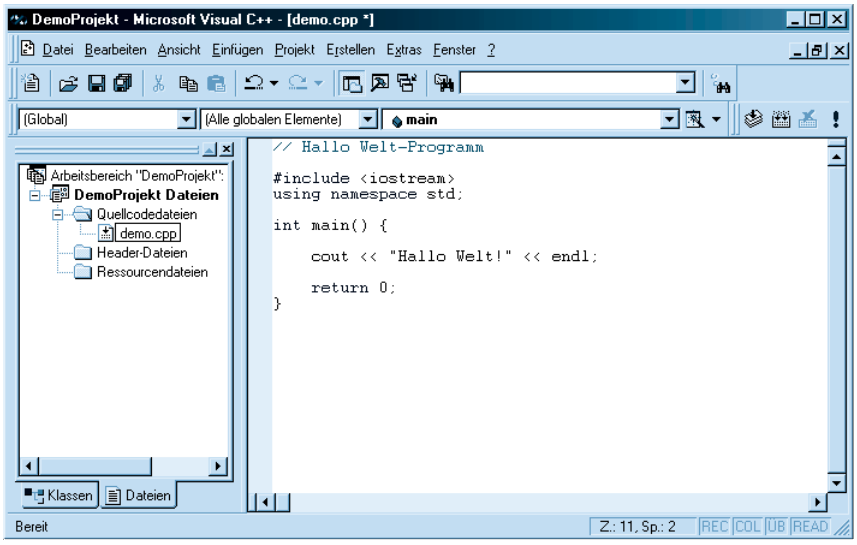


Abbildung 3.4: Der Programmcode wurde in die Datei demo.cpp *eingetippt.*

Speichern und kompilieren

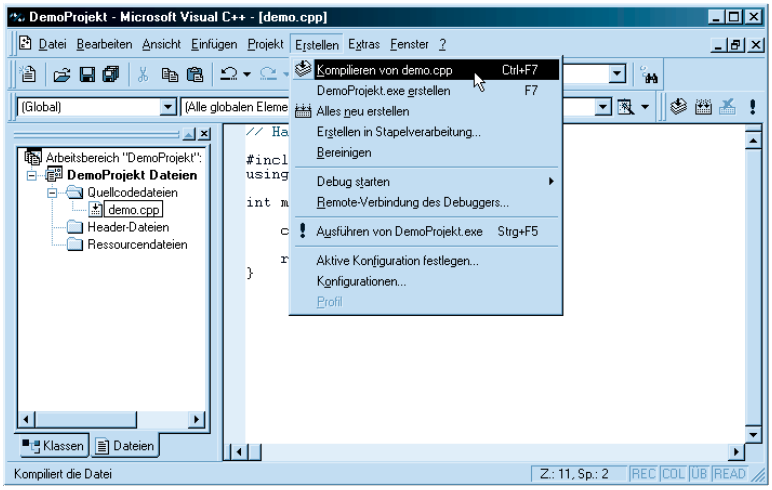
Bevor wir das Programm kompilieren, speichern wir den Quellcode ab. Dies ist zwar nicht unbedingt notwendig, gibt uns aber das sichere Gefühl, dass unser Programmcode nicht verloren geht, wenn die Visual C++-Entwicklungsumgebung beim Kompilieren (oder späteren Ausführen) des Programms abstürzen sollte.

1 Speichern Sie die Dateien des Projekts durch Aufruf des Befehls DATEI/SPEICHERN oder durch Drücken der Tastenkombination Strg S.

> **Tipp**
>
> *Wenn die aktuell geladene Quelldatei Änderungen enthält, die noch nicht abgespeichert wurden, sehen Sie in der Titelleiste ein Sternchen neben dem Dateinamen.*

Kompilieren

Bis jetzt haben wir nicht mehr als eine ganz normale Textdatei, deren Inhalt zufälligerweise der C++-Sprachspezifikation entspricht. Das mag nicht sonderlich aufregend klingen, aber unter Umständen bedeutet es, dass wir nur noch fünf Minuten von unserem ersten eigenen Programm entfernt sind.



2 Rufen Sie den Befehl ERSTELLEN/KOMPILIEREN VON DEMO.CPP auf, um den Quelltext zu kompilieren.

Im unteren Bereich der Visual C++-IDE wird jetzt das Ausgabefenster eingeblendet, in dem Sie den Fortschritt der Kompilation verfolgen können.

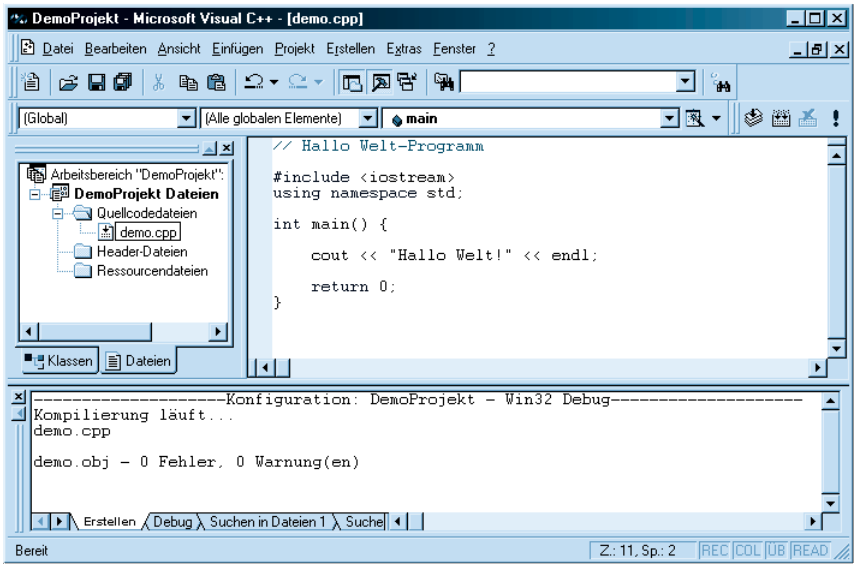


Abbildung 3.5: Null Fehler – Super!!!

Nur selten kommt es vor, dass man beim Aufsetzen des Quelltextes keinen Fehler macht. Oft sind es ganz unnötige Tippfehler, die dem C++-Novizen das Studium zur Hölle machen. Dabei sind diese so genannten »syntaktischen Fehler« noch recht harmlos, denn sie werden – im Gegensatz zu den logischen Fehlern – vom Compiler entdeckt und meist recht gut lokalisiert.

Syntaxfehler beheben

Syntaktische Fehler (im Gegensatz zu logischen Fehlern, die sich dadurch bemerkbar machen, dass das Programm nicht das tut, wofür es geschrieben wurde) werden bereits vom Compiler entdeckt und angezeigt.

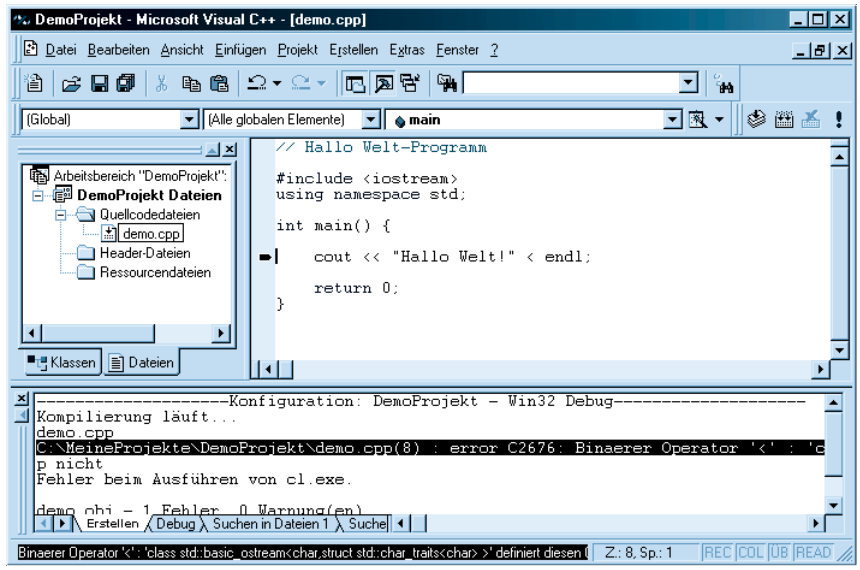
1 Bauen Sie in den Quelltext einen Fehler ein. Löschen Sie dazu beispielsweise eine der eckigen Klammern aus der `cout`-Zeile:

```
cout << "Hallo Welt!" < endl;
```

Tipp

Schneller kompiliert man durch Drücken der Tastenkombination Strg F7.

2 Rufen Sie erneut den Befehl ERSTELLEN/KOMPILIEREN VON DEMO.CPP auf.



3 Scrollen Sie die Anzeige im Ausgabefenster und doppelklicken Sie auf die Fehlermeldung.

Im Editor erscheint am linken Rand ein Pfeil, der die Zeile markiert, in der der Compiler den Fehler vermutet. Suchen Sie in der Zeile nach dem Fehler. Eventuell kann Ihnen der Text der Fehlermeldung dabei behilflich sein.

4 Korrigieren Sie den Fehler:

```
cout << "Hallo Welt!" << endl;
```

5 Drücken Sie Strg F7, um das Programm erneut kompilieren und linken zu lassen.

Die exe-Datei erstellen

Treten beim Kompilieren keine Fehler auf, übersetzt der Compiler den Quelltext in Maschinencode und speichert diesen in einer eigenen Datei mit der Extension *.obj*. Damit haben wir aber noch kein ausführbares Programm. Dazu müssen wir erst noch den Linker aufrufen, der aus dem Maschinencode in der *obj*-Datei eine ausführbare *exe*-Datei macht.

1 Rufen Sie den Befehl ERSTELLEN/DEMOPROJEKT.EXE ERSTELLEN auf. Fortschritt und Erfolg des Linkvorgangs können wir wiederum im Ausgabefenster kontrollieren.

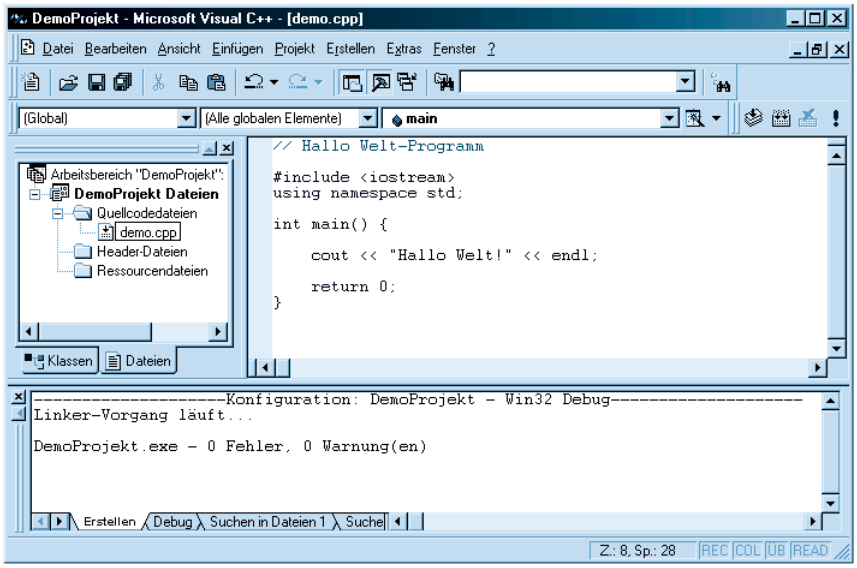
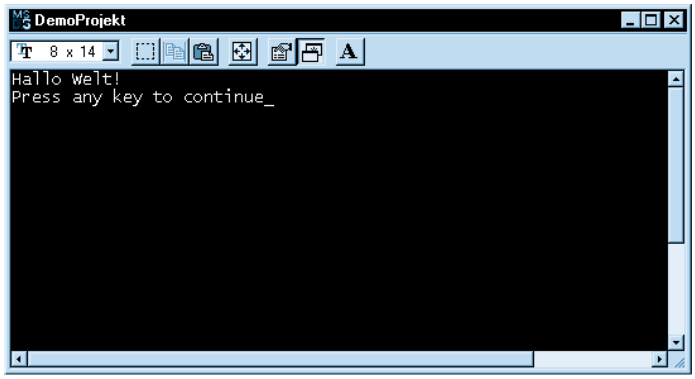


Abbildung 3.6: Das Programm wurde erfolgreich gelinkt, die exe*-Datei erzeugt.*

Das Programm testen

Wenn der Compiler das Programm ohne Probleme erstellen konnte, bedeutet dies noch nicht, dass das Programm auch korrekt arbeitet. Der letzte Schritt bei der Programmerstellung besteht daher darin, das Programm auszuführen und zu testen, ob es das macht, wofür es programmiert wurde.



1 Rufen Sie den Befehl ERSTELLEN/AUSFÜHREN VON DEMOPROJEKT.EXE auf.

Die IDE lädt das Programm und führt es in einem eigenen Konsolenfenster aus.

> **Hinweis**
>
> *Die Autorenedition von Visual C++ erweitert jedes Programm automatisch um ein Meldungsfenster, das beim Start des Programms erscheint und den Anwender darauf hinweist, dass das Programm nicht mit einem lizensierten Compiler erstellt wurde. Wenn Sie Ihre Programme kommerziell vertreiben wollen, müssen Sie sich eine Vollversion des Compilers kaufen.*

Im Falle unseres Beispielprogramms gibt es nicht viel zu testen. Sie müssen lediglich schauen, ob die Ausgabe korrekt ist.

Die Aufforderung »Press any key to continue« (Fortfahren mit beliebiger Taste) ist übrigens nicht Teil unseres Programms. Sie wird von Visual C++ erzeugt und verhindert, dass das Konsolenfenster automatisch geschlossen wird, nachdem das Programm beendet wurde – so haben wir Zeit, uns die Ausgaben des Programms in Ruhe anzuschauen

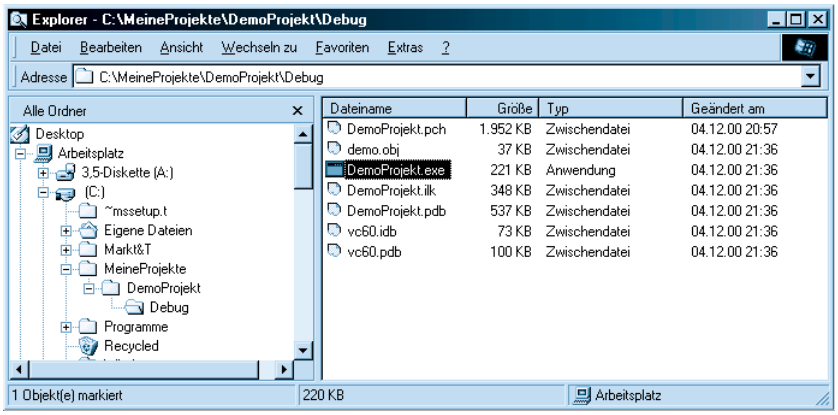
2 Drücken Sie eine beliebige Taste um das Konsolenfenster zu schließen.

Tipp

Wenn Sie Änderungen oder Korrekturen am Quelltext vornehmen, brauchen Sie danach nicht jedes Mal brav hintereinander die Befehle zum Kompilieren, Linken und Ausführen aufzurufen. Rufen Sie nur den letzten Befehl zum Ausführen auf (oder drücken Sie die Tastenkombination Strg F5 *). Visual C++ erkennt automatisch, dass geänderte Quelldateien vorliegen und fragt Sie mit einem Meldungsfenster, ob diese neu erstellt werden sollen. Wenn Sie dies bestätigen, wird der Quellcode neu kompiliert, gelinkt und ausgeführt.*

Das Programm ausführen

Natürlich will man nicht immer Visual C++ aufrufen, um ein selbst erstelltes Programm auszuführen. Aber das ist ja auch nicht nötig. Die *exe*-Datei des Programms ist auf der Festplatte im Verzeichnis des Projekts abgespeichert (Unterverzeichnis *Debug*) und kann direkt aufgerufen werden.



1 Öffnen Sie im Windows Explorer das Verzeichnis des Projekts und wechseln Sie in das Unterverzeichnis *Debug* (dieses wurde automatisch von Visual C++ angelegt).

2 Doppelklicken Sie auf die *exe*-Datei.

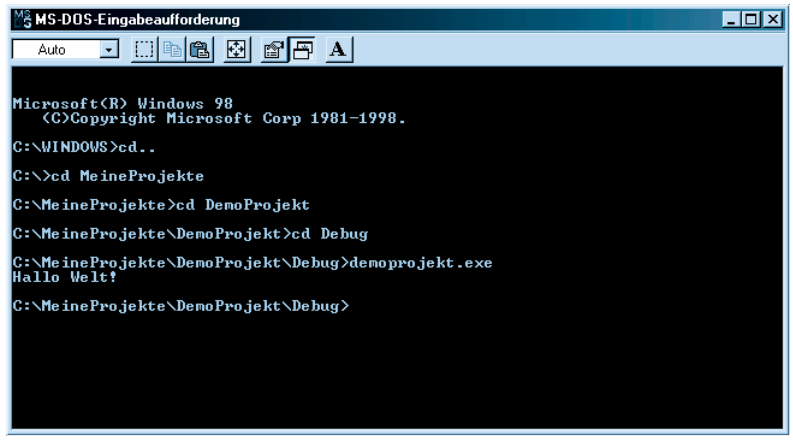
Hinweis

Erschrecken Sie nicht wegen der riesigen Dateien im Debug*-Verzeichnis Ihres Projekts* (DemoProjekt.pch, DemoProjekt.pdb *etc.) Dies sind lediglich Hilfsdateien, die das Kompilieren und Linken beschleunigen. Wenn das Programm fertig und ausgetestet ist, können Sie alle Dateien mit den Endungen* pch, obj, ilk, pdb *und* idb *löschen.*

Leider sieht man das Konsolenfenster mit dem Programm nur kurz aufflackern. Woran liegt das? Konsolenprogramme verfügen wie gesagt über keine eigenen Fenster. Stattdessen schicken sie ihre Ausgaben an eine Konsole. Aus diesem Grund richtet Windows für jedes Konsolenprogramm, das unter Windows ausgeführt wird, ein eigenes Konsolenfenster ein. Das Problem ist nur, dass Windows das Konsolenfenster automatisch schließt, wenn das Programm fertig ist. Unser Beispielprogramm ist besonders schnell fertig (es gibt ja lediglich einen kurzen Text aus) und darum wird das Konsolenfenster sofort wieder geschlossen.

Eine bessere Lösung zur Ausführung von Konsolenprogrammen ist daher meist der Aufruf von der MS-DOS-Eingabeaufforderung.

1 Rufen Sie die MS-DOS-Eingabeaufforderung auf. Öffnen Sie dazu das START-Menü von Windows und wählen Sie den Eintrag PROGRAMME/MS-DOS-EINGABEAUFFORDERUNG auf.



2 Wechseln Sie mit Hilfe des `cd`-Befehls in das Verzeichnis, in der die *exe*-Datei steht.

3 Rufen Sie das Programm von der Konsole aus auf.

> **Hinweis**
>
> *Mit Hilfe des Befehls* `dir` *oder* `dir /p` *können Sie sich den Inhalt des aktuellen Verzeichnisses anzeigen lassen.*

Visual C++ beenden

Nach getaner Arbeit wird aufgeräumt. Zuerst schließen wir das Projekt, genauer gesagt, den Arbeitsbereich unseres Projekts.

Arbeitsbereiche schließen

Wenn Sie mit der Arbeit an Ihrem Programm fertig sind oder ein neues Projekt beginnen wollen, schließen Sie den Arbeitsbereich des aktuellen Projekts.

1 Rufen Sie den Befehl DATEI/ARBEITSBEREICH SCHLIESSEN auf.

2 Lassen Sie alle Dokumentfenster schließen.

3 Lassen Sie ungespeicherte Änderungen gegebenenfalls sichern.

Visual C++ schließen

Wenn Sie Ihre Arbeit mit Visual C++ ganz beenden wollen ...

1 ... rufen Sie den Befehl DATEI/BEENDEN auf.

Bestehende Projekte neu öffnen

Wenn Sie am nächsten Tag an Ihrem Projekt weiterarbeiten wollen, rufen Sie zuerst Visual C++ und laden dann das gewünschte Projekt.

Es gibt mehrere Möglichkeiten, um ein bestehendes Programm wieder zu öffnen:

- Sie können den Befehl DATEI/ARBEITSBEREICH ÖFFNEN aufrufen (denken Sie daran, dass Visual C++ Projekte in Arbeitsbereichen organisiert) und die *dsw*-Datei aus dem Projektverzeichnis laden. (So können Sie auch die Projekte auf der Buch-CD öffnen.)

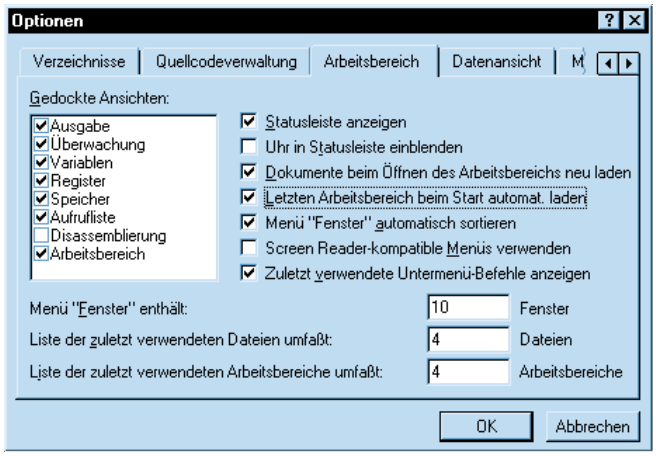
- Die zuletzt bearbeiteten Arbeitsbereiche können über den Befehl DATEI/ZULETZT GEÖFFNETE ARBEITSBEREICHE geladen werden.
- Sie können die IDE so einstellen, dass beim Start von Visual C++ automatisch der zuletzt bearbeitete Arbeitsbereich geladen wird.

Letztes Projekt automatisch beim Start laden

Wenn Sie möchten, können Sie Visual C++ so einrichten, dass beim Start von Visual C++ automatisch der Arbeitsbereich des zuletzt bearbeiteten Programms geladen wird.

1 Rufen Sie den Befehl EXTRAS/OPTIONEN auf und wählen Sie im Dialogfenster OPTIONEN das Register ARBEITSBEREICH aus.

Um zum Register ARBEITSBEREICH zu wechseln, müssen Sie zuerst die Anzeige der Register verrücken, bis der Reiter ARBEITSBEREICH sichtbar wird. Klicken Sie dazu auf die Pfeiltaste in der Register-Zeile.



2 Aktivieren Sie die Option LETZTEN ARBEITSBEREICH BEIM START AUTOMAT. LADEN.

3 Klicken Sie auf OK.

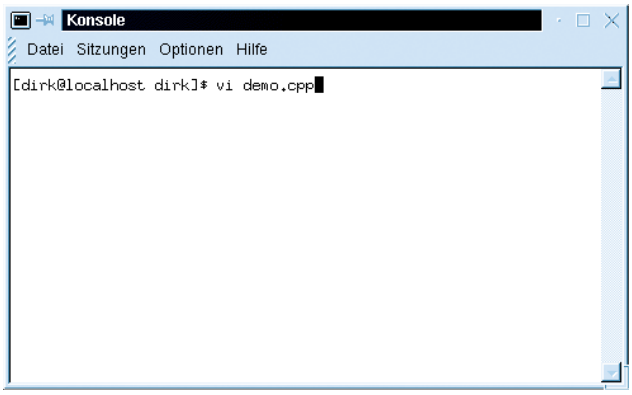
Programmerstellung mit dem g++-Compiler

Als Beispiel für einen Kommandozeilen-Compiler sei hier die Programmerstellung mit dem g++-Compiler von Linux beschrieben.

Die Verwendung eines Kommandozeilen-Compilers scheint wegen der entfallenden Projektverwaltung auf den ersten Blick einfacher als die Arbeit mit einer integrierten Entwicklungsumgebung. Das täuscht aber. Zum einen bedarf es meist nur einer kurzen Eingewöhnungszeit, bis man sich mit einer integrierten Entwicklungsumgebung vertraut gemacht hat, zum anderen haben integrierte Entwicklungsumgebungen gegenüber Kommandozeilenversionen eine Reihe von Vorteilen, die allerdings erst bei der Erstellung komplexerer Programme richtig zur Geltung kommen.

1 Öffnen Sie ein Konsolenfenster.

Wie Ihr Konsolenfenster aussieht und mit welchem Befehl es aufgerufen wird, hängt von Ihrer Linux-Version und dem verwendeten Window-Manager ab. Unter KDE können Sie Konsolenfenster beispielsweise über die KDE-Taskleiste aufrufen (Symbol TERMINAL EMULATION).

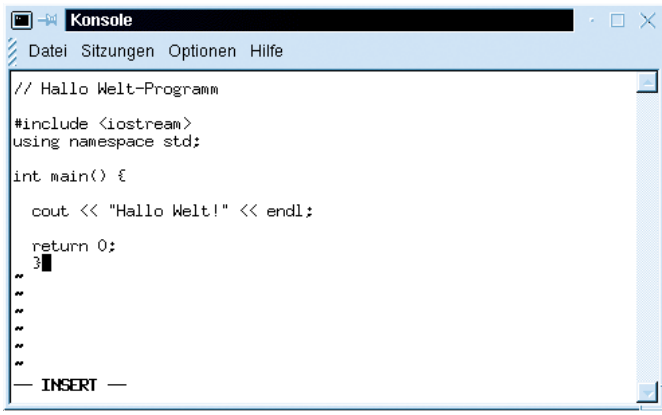


2 Legen Sie mit dem Editor *vi* eine neue Quelltextdatei an.

Statt des *vi* können Sie auch jeden beliebigen anderen ASCII-Editor verwenden, beispielsweise den *emacs* oder *KEdit* unter KDE.

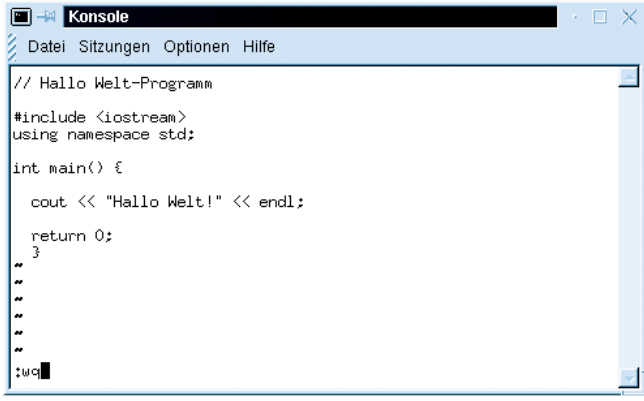
Wenn Sie den *vi* verwenden, geben Sie im Aufruf gleich den Namen der neu anzulegenden Quelldatei an. Nach dem Aufruf können Sie den Text der neuen Datei direkt im Konsolenfenster (in dem jetzt der *vi* ausgeführt wird) aufsetzen.

3 Drücken Sie die i-Taste, um in den Einfügemodus zu wechseln.

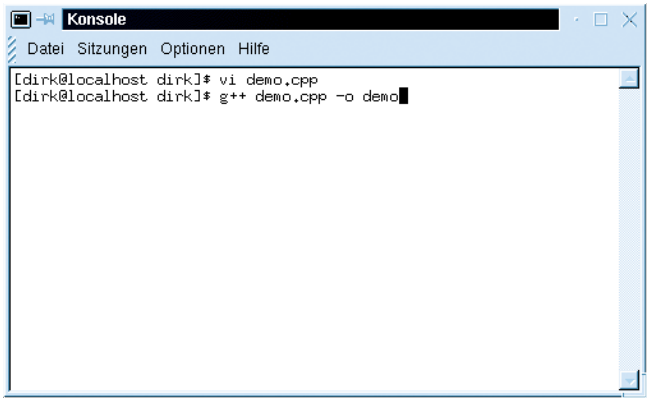


4 Geben Sie den Programmquelltext ein.

5 Drücken Sie die Esc-Taste, um in den Befehlsmodus zu wechseln.

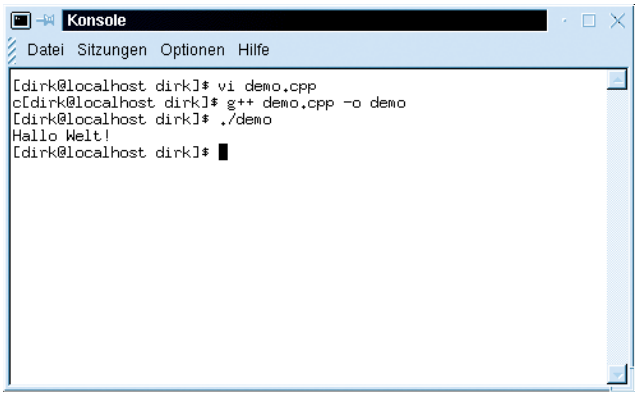


6 Tippen Sie :wq ein, um die Datei zu speichern und den *vi* zu beenden.



7 Rufen Sie von der Konsole aus den g++-Compiler auf.

Übergeben Sie dem g++-Compiler in der Kommandozeile den Namen der zu kompilierenden Datei sowie den Schalter `-o` mit dem gewünschten Namen für die ausführbare Datei.



8 Führen Sie das Programm aus.

> **Hinweis**
>
> *Die Beispiele auf der Buch-CD wurden alle mit Visual C++ erstellt und sind daher in Projekte eingebettet. Wenn Sie eines der Beispiele mit dem g++-Compiler kompilieren möchten, kopieren Sie einfach das betreffende Projektverzeichnis von der Buch-CD auf Ihre Festplatte, wechseln Sie in das kopierte Projektverzeichnis und kompilieren Sie die* cpp*-Datei des Projekts.*