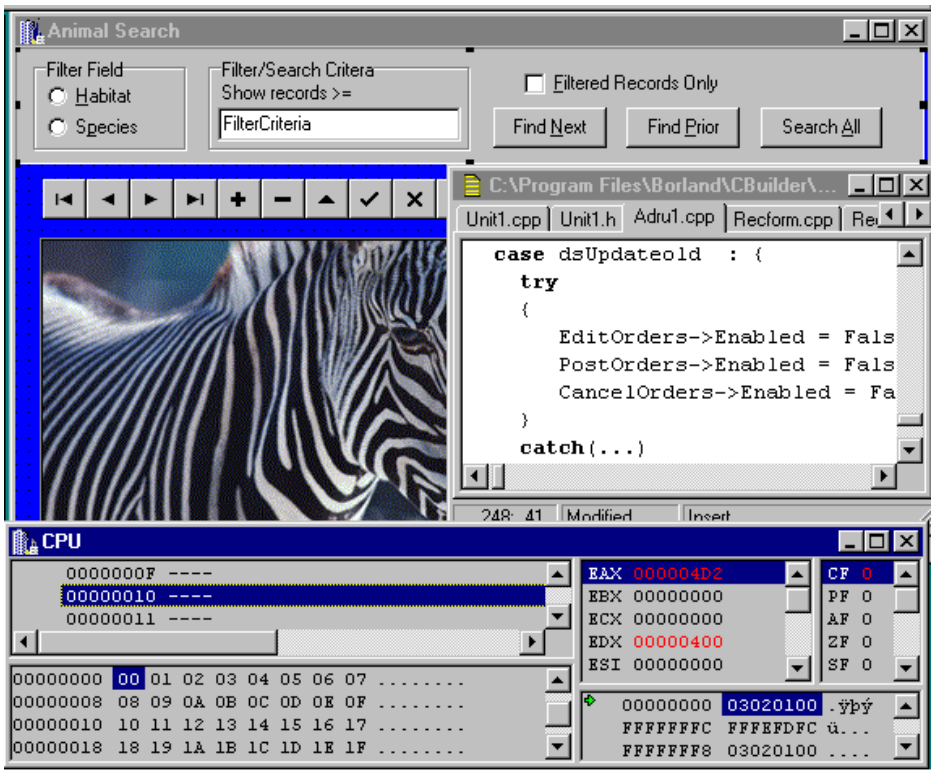
Borland C++Builder *Jump Start!*

Schnelleinstieg

Borland C++Builder für Windows 95 und Windows NT, das bedeutet die Geschwindigkeit visueller Drag&Drop-Entwicklung, die Produktivität von über 100 wiederverwendbaren Komponenten mit Quelltext und die Flexibilität skalierbarer Datenbank-Tools, kombiniert mit der Leistungsfähigkeit von C++.



C++Builder verbindet die Geschwindigkeit der visuellen Entwicklung und die Produktivität von Komponenten mit der Leistungsfähigkeit von C++.

C++Builder ist eine RAD-Entwicklungsumgebung (Rapid Application Development), mit der Sie stets die Nase vorn haben. Sie gelangen mit Ihren Anwendungen rasch vom Prototyp zum fertigen Produkt. Um Ihnen die Leistungsfähigkeit und Flexibilität von C++Builder näherzubringen, werden wir im nächsten Abschnitt eine einfach Anwendung erzeugen. Die folgenden Werkzeuge bilden den Kern der Entwicklung unter C++Builder:

- ***Komponentenpalette*** - Über 100 wiederverwendbare Komponenten, die Sie in Ihren Anwendungen einsetzen können. C++Builder Professional wird mit dem vollständigen Quelltext aller VCL-Komponenten ausgeliefert.
- ***Objektablage*** - Eine zentrale Ablage für alle wiederverwendbaren Objekte wie z.B. Formulare und Datenmodule. Durch die gemeinsame Benutzung dieser Objekte kann die Entwicklung einer Anwendung beschleunigt werden.

- ***Objektinspektor*** - Hier können Sie die Eigenschaften der verschiedenen Komponenten visuell festlegen, ohne auch nur eine Zeile Quelltext zu schreiben. Sie sehen die Ereignisse der Komponenten, die Sie mit bestimmtem Quelltext verknüpfen können, der beim Eintreten eines Ereignisses ausgeführt wird.
- ***Formularentwurfsbereich*** - In den Formularen können Sie die Benutzeroberfläche Ihrer Anwendung erstellen.
- ***Quelltexteditor*** - Ein Editor, in dem Sie Ereignisbehandlungsroutinen erzeugen und Quellcode schreiben können.
- ***Datenbank-Explorer/Data Dictionary*** - Hier können Sie Ihre Datenbanken und Tabellen durchsehen und für Datenintegrität sorgen.
- Die leistungsfähige ***CPU-Ansicht*** von C++Builder besteht aus fünf getrennten Bereichen, die Ihnen einen tiefen Einblick in die Low-Level-Aspekte einer laufenden Anwendung ermöglichen. Hier werden der disassemblierte Code, der Aufruf-Stack, die Flags, die Prozessor-Register und ein Speicherauszug angezeigt.

Eine einfache Anwendung

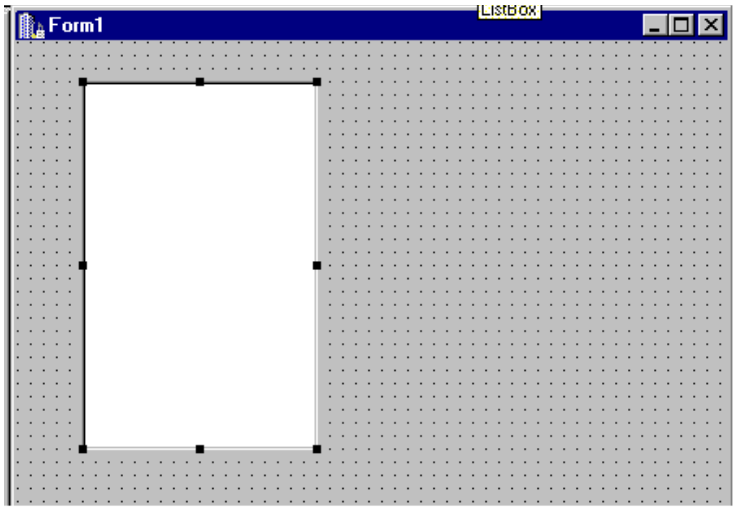
➔ *Ein solcher Pfeil kennzeichnet einzelne Arbeitsschritte, die Sie direkt in C++Builder nachvollziehen können.*

Anwendungen werden in C++Builder visuell erstellt. Das heißt, Sie wählen die gewünschten Komponenten einfach aus der *Komponentenpalette* aus. Jedes dieser Elemente, wie etwa eine Schaltfläche, besitzt eine Reihe von Eigenschaften, mit denen das Aussehen und das Verhalten vorgegeben wird. Zu jeder Komponente gehören auch mehrere Ereignisse, welche die Reaktion der Komponente auf bestimmte Vorgänge definieren. In einer ersten Beispielanwendung lernen Sie drei Komponenten kennen: ein Eingabefeld, ein Listenfeld und eine Schaltfläche. Die Schaltfläche soll dazu dienen, Einträge in die Liste einzufügen.

➔ Starten Sie jetzt C++Builder, und wählen Sie *Datei/Neue Anwendung*.

➔ Klicken Sie die verschiedenen Register am oberen Rand der Komponentenpalette an.

Beachten Sie beim Wechsel von der Registerkarte *Standard* zu *Win95*, daß sich die verfügbaren Komponenten ändern (eine vollständige Liste der Komponenten finden Sie im nächsten Abschnitt). Plazieren Sie jetzt die erste Komponente im Formular.



In der visuellen Entwicklungsumgebung von C++Builder können mit Hilfe wiederverwendbarer Komponenten Anwendungen sehr schnell erstellt werden.

➔ **Öffnen Sie die Registerkarte *Standard*.**

➔ **Klicken Sie auf die Komponente *ListBox* .**

➔ **Klicken Sie auf eine beliebige Stelle im Formularentwurfsbereich, um das Listenfeld zu plazieren.**

➔ **Klicken Sie auf das Symbol und plazieren Sie ein Eingabefeld im Formular.**

➔ **Klicken Sie auf die Komponente und plazieren Sie eine Schaltfläche im Formular.**

Echte visuelle Entwicklung und leistungsfähiges C++

Vollständig integriert in die visuellen Entwicklungsmodule bietet C++Builder umfangreiche Kommandozeilenprogramme, einen CPU-Monitor und die mehrfach ausgezeichnete Borland C++ Compiler-Technologie mit inkrementellem Smart-Link-Vorgang. Daraus entstehen hohe Flexibilität und kurze Compile-Zeiten, die von professionellen Programmierern erwartet werden.

Sie können im Quelltexteditor ANSI C++ Code schreiben und compilieren und dabei die neuesten ANSI-Merkmale nutzen: Templates, Exceptions, RTTI (Runtime Type Information = Laufzeit-Typinformationen) und Namespaces. Die Entwicklung wird durch die Standard C++ Bibliothek vereinfacht. Dort finden Sie verschiedenste Klassen und die STL-Bibliothek (Standard Template Library) mit Container- und Iterator-Klassen. Zusätzlich enthalten sind die neuen ANSI/ISO C++ Erweiterungen, wie bool, explicit, mutable und typename.

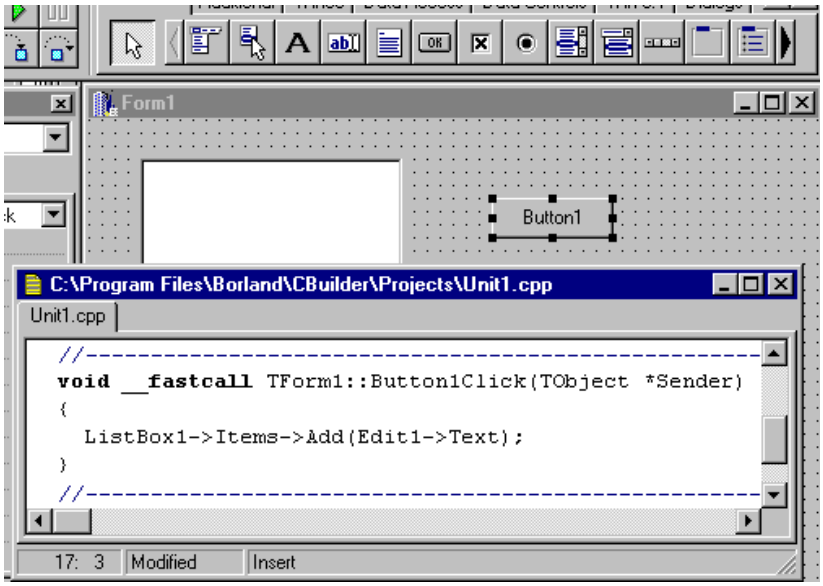
Eigenschaften, Methoden und Ereignisse

Das Kürzel RAD steht für Rapid Application Development und bedeutet, daß Objekteigenschaften, -methoden und -ereignisse unterstützt werden. **Eigenschaften** ermöglichen Ihnen, auf einfache Weise Komponentenmerkmale wie Titel, Hilfekontext oder Datenquelle festzulegen. **Methoden** ermöglichen die Ausführung von Aktionen, wie etwa das Abspielen von Medien über ein Multimedia-Steuerelement. Mit **Ereignissen** kann auf Windows-Botschaften reagiert werden, die beispielsweise den Status einer Maustaste oder die Auswahl eines Steuerelements signalisieren. Ebenso können Ereignisse auch auf benutzerdefinierte Statusänderungen, wie Datenaktualisierungen für datensensitive Steuerelemente reagieren. Dieses Konzept ermöglicht eine stabile und intuitive Entwicklungsumgebung für die Programmierung unter Windows.

- ➔ **Klicken Sie im Objektinspektor auf die Seite *Ereignisse*, damit die Ereignisse des aktuellen Objekts zu sehen sind.**
- ➔ **Doppelklicken Sie auf die Schaltfläche im Formular.**
- ➔ **Geben Sie anschließend im Quelltexteditor folgende Anweisung ein:**

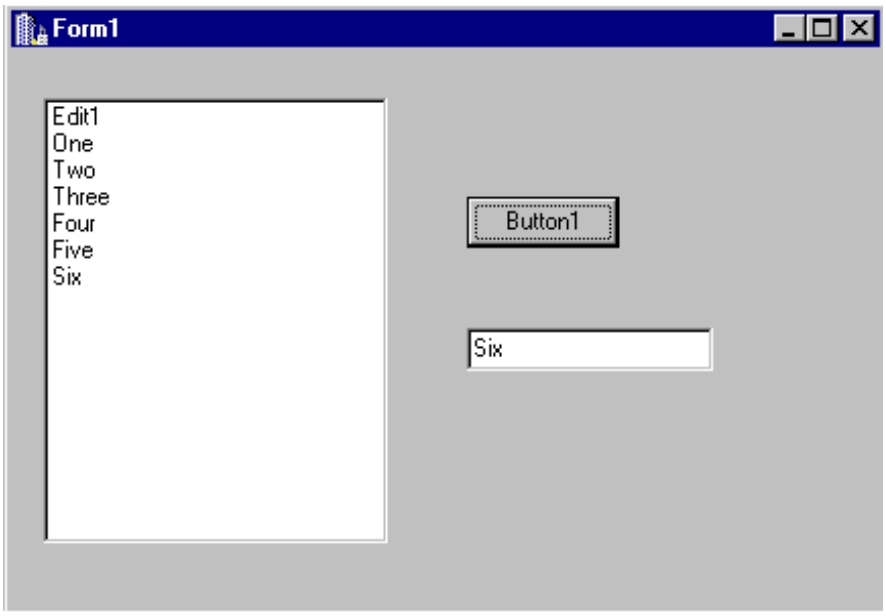
```
ListBox1->Items->Add(Edit1->Text);
```

Damit weisen Sie das Objekt (ListBox1) an, seiner Eigenschaft *Items* mit der Methode *Add* ein neues Element hinzuzufügen. Als neue Einträge gelten die Werte, die zur Laufzeit in die Eigenschaft *Text* der Komponente *EditBox* eingegeben werden.



Der Quelltext wird in die Datei Unit1.cpp eingegeben—C++Builder compiliert beliebigen ANSI C++ Code.

- ➔ **Klicken Sie auf den grünen Pfeil  , um die Anwendung zu compilieren und zu linken.**
- ➔ **Sobald die Anwendung läuft, klicken Sie auf Button1, um den Text, der im Eingabefeld steht, in die Liste einzufügen.**



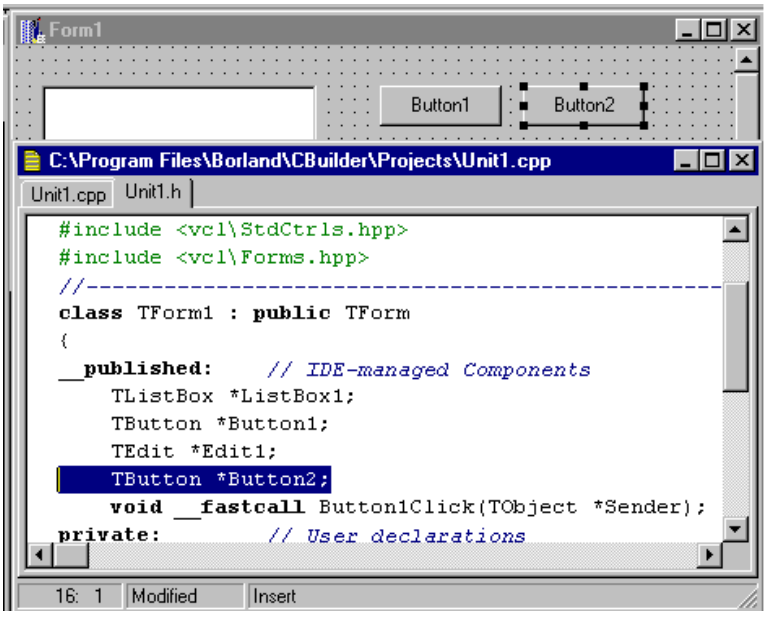
Produzieren Sie schnelle C++ Programme zur uneingeschränkten Distribution

Borland Two-Way-Tools

In C++Builder haben Sie stets direkten Zugriff auf den Quelltext. Die Borland Two-Way-Tools ermöglichen die Arbeit am Code auf zwei Arten, denn die visuelle Entwurfsumgebung und der Quelltexteditor sind nahtlos integriert. Sie können jederzeit zwischen dem Editor und den visuellen Werkzeugen wechseln und trotzdem sicher sein, daß beide synchron arbeiten.

So sehen Sie die Two-Way-Tools in Aktion:

- ➔ **Klicken Sie mit der rechten Maustaste in Unit1.cpp, und wählen Sie *Umschalten Formular/Unit*.**
- ➔ **Teilen Sie den Bildschirm so auf, daß Sie das Formular und die Header-Datei zugleich sehen können.**
- ➔ **Plazieren Sie ein Objekt (z. B. eine Schaltfläche) im Formular.**



C++ Builder errichtet keine Barrieren zwischen Ihnen und Ihrem Quelltext.

Beachten Sie, daß der entsprechende Code sofort in der Header-Datei erscheint, sobald Sie das Objekt im Formular erstellen. Diese nahtlose Interaktion zwischen der visuellen Umgebung und dem Quelltexteditor ermöglicht C++ Entwicklern, rasch visuelle Anwendungen zu erstellen und dennoch die vollständige Kontrolle über den zugrundeliegenden Quelltext zu behalten.

Jede mit C++Builder erstellte Anwendung beginnt mit den folgenden drei Dateien:

- **Unit1.cpp:** CPP-Dateien enthalten die eigentliche Implementierung der Anwendung. In dieser Datei schreiben Sie die verschiedenen Ereignisbehandlungsroutinen. Diese Routinen steuern, wie die Anwendung auf Aktionen reagiert, die vom Benutzer oder der Anwendung stammen.
- **Unit1.h**: Die Header-Datei enthält die Konstruktoren oder Deklarationen für die einzelnen Objekte. Das Fastcall-Konzept ist ein intelligentes Verfahren, Parameter zu übergeben. Dabei erfolgt die Übergabe per Registrierung und nicht durch den Speicher, da Speicheraufrufe stets neu geladen und ausgeführt werden müssen.
- **Project1.cpp**: Die Projektdatei verwaltet alle Objekte Ihrer Anwendung. Jedes neue Formular, jede Unit und jedes Datenmodul wird in die Projektdatei aufgenommen. Den zugehörigen Quelltext können Sie mit *Ansicht/Projekt-Quelltext* anzeigen.

Alle diese Dateien arbeiten mit vorcompilierten Headern. Dabei handelt es sich um binäre Dateien, die schnell zu laden sind und bereits eine erste Parsing-Stufe absolviert haben. Der Zeitbedarf einer Compilierung sinkt dadurch drastisch. Selbstverständlich können alle Compiler-Direktiven und Include-Dateien vom Benutzer beliebig angepaßt werden.

Borland-Dienstprogramme

In C++Builder wird die Leistungsfähigkeit von C++ mit einer Gruppe von objektorientierten Tools zur schnellen Anwendungsentwicklung kombiniert. C++Builder ergänzt vorhandene Visual C++ und Borland C++ Anwendungen durch schnelle, produktive Front-Ends. C++Builder schützt Ihre Investitionen in bestehenden C++ Code, da sämtlicher ANSI C++ Code verarbeitet werden kann.

Borland C++ und Borland C++Builder

Nutzen Sie die Verknüpfung von Borland C++ und C++Builder zur schnellen Entwicklung von Benutzeroberflächen. In kürzester Zeit erweitern Sie Ihre C++ Anwendungen, denn beide Produkte basieren auf demselben Compiler und inkrementellen Linker und verfügen über eine lückenlose technische Dokumentation. C++Builder-Formulare können leicht in OWL- und MFC-Anwendungen integriert werden. Umgekehrt lassen sich mit Borland C++ DLLs, OBJs oder COM-Objekte für C++Builder-Anwendungen produzieren. Als Zusatz bietet C++Builder die vollständige Unterstützung des Industriestandards. Dies umfaßt ANSI C++, die Win32-API, ActiveX, OLE-Automatisierung, ODBC, DCOM, MAPI, Unicode, WinSock, ISAPI und NSAPI.

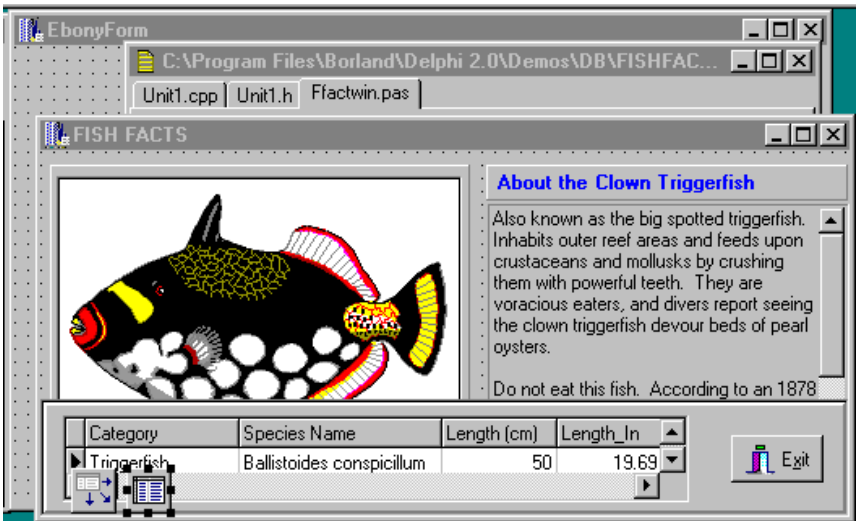
Delphi und C++Builder

Auch für Delphi-Entwickler gibt es Gründe, C++ zu verwenden. C++Builder für Windows 95 und NT ist die perfekte Ergänzung zu Ihren Delphi-Anwendungen. Sie können auf Ihr erworbenes Delphi-Know-how aufbauen, indem Sie dieselbe Bibliothek visueller Komponenten (VCL), dieselbe intuitive IDE, die bekannten Two-Way-Tools, die Visuelle Formularvererbung und die skalierbare Datenbankanbindung genau wie in Delphi nutzen. Ihr *Delphi-Code, Ihre Komponenten, Datenmodule und Formulare* können in C++Builder-Anwendungen wiederverwendet werden. Delphi bleibt weiterhin am einfachsten einsetzbar und bietet die höchste Produktivität, während C++Builder die ganze Leistungsfähigkeit und Systemnähe von C++ all denen zugänglich macht, die an Delphi die unvergleichliche Entwicklungsgeschwindigkeit schätzen. In größeren Organisationen kann jeder die Sprache seiner Wahl

benutzen, und trotzdem können Objekte und Code problemlos ausgetauscht werden. Das Ergebnis ist eine bisher unbekannte Produktivität über alle Abteilungen hinweg.

So fügen Sie ein vorhandenes Objekt in Ihre C++Builder-Anwendung ein:

- ➔ **Wählen Sie *Datei/Zum Projekt hinzufügen*.**
- ➔ **Ändern Sie die Eigenschaft *Name* Ihres C++Builder-Formulars in *CPPForm*. (Dadurch werden Namenskonflikte vermieden.)**
- ➔ **Wechseln Sie in ein Verzeichnis, das ein Delphi-Formular enthält: PROGRAMME/BORLAND/DELPHI30/DEMOS/DB/FISHFACT**
- ➔ **Wählen Sie in der Dropdown-Liste die Anzeige von PAS-Dateien. Klicken Sie auf *OK*.**
- ➔ **Kehren Sie mit F12 zu Form1 zurück.**



In C++Builder können mit Delphi erzeugte Formulare, Datenmodule, Komponenten und Quelltexte wiederverwendet werden.

Nun ist das Delphi-Formular Teil Ihrer Anwendung. Sie müssen noch einen C++ Header für das Formular generieren.

- ➔ **Klicken Sie auf ein beliebiges CPP-Formular in der Anwendung, und wählen Sie *Datei|Unit Hdr. einfügen...***

Die Anwendung kann jetzt compiliert und gestartet werden. Auf Grundlage der Visuellen Formularvererbung können Sie das Delphi-Formular mit C++ verändern.

Die Komponentenpalette

Borland C++Builder wird mit der Borland VCL (Visual Component Library) ausgeliefert, die mehr als 100 wiederverwendbare Komponenten enthält, mit denen leistungsfähige Windows-Anwendungen per Drag&Drop erstellt werden können. In der VCL sind sämtliche Elemente der Benutzeroberfläche von Windows 95 (Wippregler, Fortschrittsanzeige, Schieberegler, Statusleiste, RTF-Eingabefeld usw.) enthalten. Weitere Komponenten ergänzen die vorhandenen Dialogobjekte, datensensitiven Elemente und Multimedia-Tools, damit Sie komplexe Anwendungen entwickeln und schnell vom Prototyp zum fertigen Produkt gelangen können.

Damit professionelle Entwickler schnell und auf einfache Weise wiederverwendbare Komponenten erzeugen können, enthalten die C++Builder-Versionen Professional und Client/Server Suite den vollständigen Quelltext der Borland-VCL, der einen tiefen Einblick in die Arbeitsweise von C++Builder und in die Komponenten selbst ermöglicht. Mit Hilfe dieses Quelltextes können Sie aus Beispielen lernen und vorhandene Komponenten wiederverwenden.

Mit C++Builder steht Ihnen die Leistungsfähigkeit der objektorientierten Programmierung zur Verfügung, mit der Sie robuste und effiziente Anwendungen entwickeln können. Erzeugen Sie Ihre eigenen Komponenten mit der bewährten objektorientierten Architektur von C++Builder, die auch die nahtlose Integration von ActiveX-Steuerelementen ermöglicht. Sie können für Ihre Anwendung vorhandene Komponenten verwenden, selbst welche definieren oder von bestehenden Komponenten neue ableiten − alles innerhalb der Entwicklungsumgebung von C++Builder.

Die in C++Builder verwendeten Komponenten bauen auf verschiedenen Objekttypen auf, die das Grundgerüst einer Anwendung bilden. Dieses Grundgerüst ist die Bibliothek visueller Komponenten (VCL), in der das in C++Builder implementierte Objektmodell verwendet wird, um dem Entwickler ein durch und durch objektorientiertes Entwicklungssystem zur Verfügung zu stellen.

Die VCL enthält sämtliche Elemente der Benutzeroberfläche von Windows 95 und viele weitere Komponenten. Dazu gehören alle von Windows-Programmierern benötigten Standardkomponenten wie Eingabefelder, Kombinationsfelder, Listenfelder, Register usw.

Ein wichtiges Merkmal von C++Builder ist die Möglichkeit, Komponenten in einer visuellen Entwicklungsumgebung nicht nur zu *verwenden*, sondern gegebenenfalls auch neu zu *erzeugen*. Neue Komponenten können durch die Vererbungsmöglichkeiten der objektorientierten Programmierung entweder von vorhandenen abgeleitet und um zusätzliche Funktionen erweitert oder vollständig neu definiert werden.

Die Möglichkeit, neue Komponenten zu erzeugen, bedeutet für den Entwickler, daß er nicht die Umgebung zu wechseln braucht, wenn er andere Steuerelemente benötigt. Das Entwerfen und Implementieren neuer Komponenten ist in C++Builder kein Problem. Entwickler können die Standardeinstellungen eines bestimmten Steuerelements ändern, vorhandene Komponenten mit neuen Funktionen versehen, Windows-Steuerelemente von Fremdherstellern kapseln oder selbst völlig neue Komponenten erzeugen.

So erzeugen Sie neue C++Builder-Komponenten:

- Leiten Sie sie von einem vorhandenen Komponententyp ab.
- Definieren Sie neue Felder, Eigenschaften und Methoden.

- Registrieren Sie die neue Komponente.

Die Komponenten in der Registerkarte Standard

Die Registerkarte *Standard* enthält die Windows-Standardkomponenten.

TMainMenu	Das Hauptmenü eines Formulars.
TPopUpMenu	Ein Popup-Menü in einem Formular.
TLabel	Text, der nicht geändert werden kann (z.B. ein Titel).
TEdit	Ein Eingabebereich, in dem der Benutzer eine Datenzeile eingeben oder ändern kann.
TMemo	Ein Eingabebereich, in dem der Benutzer mehrere Datenzeilen eingeben oder ändern kann.
TButton	Ein Schalter, mit dem der Benutzer eine Aktion ausführen kann.
TCheckBox	Ein Steuerelement, das der Benutzer aktivieren oder deaktivieren kann, um zwischen Ja/Nein bzw. True/False zu wählen.
TRadioButton	Ein Steuerelement, das der Benutzer aktivieren oder deaktivieren kann, um zwischen Ja/Nein bzw. True/False zu wählen (wird immer in Gruppen verwendet).
TListBox	Eine bildlauffähige Liste von Auswahlmöglichkeiten.
TComboBox	Eine Liste von Auswahlmöglichkeiten in einem kombinierten Eingabe- und Listenfeld. Der Benutzer kann Daten in das Eingabefeld eingeben oder einen Eintrag aus der Liste wählen.
TScrollBar	Ein Steuerelement, mit dem der sichtbare Bildausschnitt einer Liste oder eines Formulars verschoben werden kann.
TGroupBox	Ein Container für eine Gruppe zusammengehöriger Optionen.
TRadioGroup	Ein Gruppenfeld für Optionsfelder.
TPanel	Ein Container für andere Komponenten, der als Werkzeug- oder Statusleiste verwendet werden kann.

Die Komponenten in der Registerkarte Zusätzlich

Die Registerkarte *Zusätzlich* der Komponentenpalette enthält weitere Windows-Steuerelemente.

TBitButton	Ein Schalter, der ein Bitmap anzeigen kann.

TSpeedButton	Ein Schalter, der eine Grafik anzeigen kann. Als Gruppe bilden solche Schalter eine Schalterleiste.
TMaskEdit	Ein Eingabefeld, das bestimmte Eingabe- und Anzeigeformate ermöglicht.
TStringGrid	Ein Gitter, das Strings in Spalten und Zeilen anzeigt.
TDrawGrid	Ein Gitter, das Daten in Spalten und Zeilen anzeigt.
TImage	Ein Steuerelement, das ein Bitmap, ein Symbol oder eine Metadatei anzeigt.
TShape	Ein Steuerelement, das geometrische Formen zeichnet (z.B. Ellipsen, Kreise, Rechtecke oder Quadrate).
TBevel	Eine Linie oder ein Rechteck mit einem dreidimensionalen oder reliefartigen Aussehen.
TScrollBox	Ein größenveränderlicher Container, der automatisch Bildlaufleisten anzeigt.

Die Komponenten in der Registerkarte Win95

Die Registerkarte *Win95* der Komponentenpalette enthält Elemente der Benutzeroberfläche, die unter Windows 95 eingeführt wurden.

TTabControl	Ein Registersatz, der TTabSet ähnelt und wie die Trennblätter in einem Notizbuch aussieht.
TPageControl	Eine Reihe von übereinanderliegenden Seiten, die ein mehrseitiges Dialogfeld bilden.
TTreeView	Eine hierarchische Liste von Elementen (z.B. Absatzüberschriften eines Dokuments, Indexeinträge oder die Dateien und Verzeichnisse eines Laufwerks).
TListView	Eine Liste, die Elemente auf verschiedene Art anzeigen kann. Die Eigenschaft ViewStyle bestimmt, ob die Elemente in Spalten mit Überschriften und Unterelementen, vertikal oder horizontal und mit kleinen oder großen Symbolen angezeigt werden.
TImageList	Ein Container für eine Gruppe von Grafiken.
THeaderControl	Eine Kopfzeile (ähnlich THeader) mit mehreren Bereichen, die zur Laufzeit neu angeordnet werden können.
TRichEdit	Ein Memo-Eingabefeld, das das RTF-Format unterstützt.
TStatusBar	Ein Fenster, in dem Statusinformationen angezeigt werden.
TTrackBar	Ein Schieberegler (wahlweise mit Teilstrichen), dessen Schieber die aktuelle Position angibt.
TProgressBar	Eine Komponente, die den Fortgang einer Operation anzeigt, indem sie sich von links nach rechts mit der Farbe füllt, die im System für Hervorhebungen verwendet wird.

TUpDown	Ein Wippregler mit zwei Pfeilschaltern. Durch Klicken auf einen der Pfeile wird der numerische Wert der Eigenschaft Position erhöht oder erniedrigt.
THotKey	Eine Komponente, mit der zur Laufzeit Tastenkürzel zugewiesen werden können. Der Benutzer kann beliebige Tastenkombinationen eingeben, die sich normalerweise aus einer Sondertaste (z.B. STRG, ALT oder UMSCHALT) und einer weiteren Taste (z.B. einer Zeichen-, Pfeil- oder Funktionstaste) zusammensetzen.

Die Komponenten in der Registerkarte Datenzugriff

Die Registerkarte *Datenzugriff* (siehe Abbildung) enthält spezielle Komponenten für den Zugriff auf Datenbanken.

TDataSource	Das Bindeglied zwischen den Komponenten TTable, TQuery, TStoredProc und datensensitiven Komponenten wie TDBGrid.
TTable	Diese Komponente ermöglicht mit Hilfe der BDE (Borland Database Engine) den Zugriff auf Datenbanktabellen. TTable ist die Schnittstelle zwischen der BDE und TDataSource.
TQuery	Mit dieser Komponente können SQL-Anweisungen an die Datenbank-Engine (BDE oder SQL-Server) gesendet werden. TQuery ist die Schnittstelle zwischen dem SQL-Server (bzw. der BDE) und TDataSource.
TStoredProc	Mit dieser Komponente können auf dem Server gespeicherte Prozeduren ausgeführt werden.
TDatabase	Diese Komponente ist für den Datenbankzugriff nicht unbedingt erforderlich. Sie bietet aber zusätzliche Steuerungsmöglichkeiten, die für Client/Server-Anwendungen von Bedeutung sind.
TSession	Diese Komponente bietet globale Kontrolle über die Datenbankverbindungen einer Anwendung. Sie wird zur Laufzeit automatisch für jede Anwendung erzeugt, die Datenbank-Steuerelemente verwendet.
TBatchMove	Diese Komponente ermöglicht Operationen mit Gruppen von Datensätzen oder ganzen Tabellen.
TUpdateSQL	Diese Komponente ermöglicht bei schreibgeschützten Datenmengen zwischengespeicherte Aktualisierungen.

Die Komponenten in der Registerkarte Datensteuerung

Die Registerkarte *Datensteuerung* (siehe Abbildung) enthält Komponenten, die den Zugriff auf die Datenbank steuern.

TDBGrid	Diese Komponente zeigt die Daten einer Tabelle oder Abfrage tabellarisch an und ermöglicht ihre Bearbeitung.
TDBNavigator	Ein Datenbanknavigator, mit dem sich der Benutzer durch die Datensätze einer Tabelle oder Abfrage bewegen und bestimmte Operationen durchführen kann (z.B. Datensätze einfügen oder löschen).
TDBText	Text in einem datensensitiven Steuerelement.
TDBEdit	Ein datensensitives Eingabefeld mit dem vollem Funktionsumfang eines normalen Eingabefeldes.
TDBMemo	Diese Komponente zeigt wie TDBEdit Text an und ermöglicht dem Benutzer, Daten in ein Feld einzugeben. TDBMemo unterstützt mehrzeiligen Text einschließlich Text-BLOBs (Binary Large Objects).
TDBImage	Diese Komponente zeigt eine in einem BLOB-Feld des aktuellen Datensatzes gespeicherte Grafik an.
TDBListbox	Ein datensensitives Listenfeld. Der Benutzer kann den Wert eines Feldes im aktuellen Datensatz ändern, indem er einen Eintrag aus der Liste wählt.
TDBCombobox	Ein datensensitives Kombinationsfeld, mit dem der Benutzer den Wert eines Feldes im aktuellen Datensatz ändern kann.
TDBCheckbox	Eine Option, die der Benutzer aktivieren oder deaktivieren kann. Diese Komponente ist mit den Daten in einem bestimmten Feld der Datenmenge verbunden.
TDBRadioGroup	Eine Gruppe von datensensitiven Optionsfeldern, von denen immer nur eines aktiviert sein kann. Diese Komponente wird für Optionen verwendet, die sich gegenseitig ausschließen.
TDBLookupList	Eine Liste mit Werten aus einer anderen Datenmenge.
TDBLookupComboBox	Eine Dropdown-Liste mit Werten aus einer anderen Datenmenge.
TDBCtrlGrid	Diese Komponente zeigt mehrere Datensätze in einem individuellen Layout an.

Die Komponenten in der Registerkarte System

Die Registerkarte *System* (siehe Abbildung) enthält spezielle System-Steuerelemente.

TTimer	Eine Komponente, die periodisch das Ereignis OnTimer auslöst.
TPaintBox	Diese Komponente ermöglicht einer Anwendung, innerhalb eines rechteckigen Bereichs des Formulars zu zeichnen, nicht aber außerhalb dieses Bereichs.
TFileListBox	Eine Liste mit den Dateien im aktuellen Verzeichnis. Um Dateien in anderen

	Verzeichnissen anzuzeigen, muß der Wert der Eigenschaft Directory geändert werden.
TDirectoryListBox	Ein Listenfeld mit der Verzeichnisstruktur des aktuellen Laufwerks.
TDriveComboBox	Ein Kombinationsfeld, das die verfügbaren Laufwerke anzeigt.
TFilterComboBox	Ein Kombinationsfeld, das verschiedene Dateifilter anbietet.
TMediaPlayer	Eine Steuerung für Geräte, die einen MCI-Treiber bereitstellen. Die Komponente besteht aus mehreren Schaltern (Wiedergabe, Stop usw.), mit denen ein Multimedia-Gerät (CD-ROM-Laufwerk, MIDI-Sequencer usw.) bedient werden kann.
TOleContainer	Diese Komponente ermöglicht einer C++Builder-Anwendung das Einbetten oder Verknüpfen von OLE-Objekten.
TDdeClientConv	Eine DDE-Konversation mit einem DDE-Server.
TDDEClientItem	Ein Element einer DDE-Konversation.
TDDEServerConv	Eine DDE-Konversation mit einem DDE-Client.
TDDEServerItem	Ein Element einer DDE-Konversation.

Die Komponenten in der Registerkarte Dialoge

Die Registerkarte *Dialoge* (siehe Abbildung) enthält verschiedene Windows-Standarddialoge, die eine einheitliche Benutzeroberfläche für Dateioperationen wie Öffnen, Speichern oder Drucken von Dateien zur Verfügung stellen.

TOpenDialog	Das Dialogfeld *Öffnen*.
TSaveDialog	Das Dialogfeld *Speichern*.
TFontDialog	Das Dialogfeld *Schrift*.
TColorDialog	Das Dialogfeld *Farbe*.
TPrintDialog	Das Dialogfeld *Drucken*, in dem der Benutzer den gewünschten Drucker, die zu druckenden Seiten, die Anzahl der Kopien und den Sortiermodus festlegen kann.
TPrinterSetupDialog	Das Dialogfeld *Druckereinrichtung*, in dem der Benutzer vor dem Drucken verschiedene Druckereinstellungen vornehmen kann.
TFindDialog	Das Dialogfeld *Suchen*.
TReplaceDialog	Das Dialogfeld *Ersetzen*. TReplaceDialog verfügt über den gesamten Funktionsumfang von TFindDialog, ermöglicht aber zusätzlich das Ersetzen des gefundenen Textes.

Datenbankanwendungen in Rekordzeit

Geschwindigkeit und Skalierbarkeit von Datenbanken

Zur Philosophie der Datenbankanwendungen in C++Builder gehört ein minimaler Zeitaufwand in der Entwurfsphase und die konsequente Anwendung eines objektorientierten Grundkonzepts sowohl bei der Benutzeroberfläche als auch im Umgang mit den Daten. Für den Datenzugriff ist die kompakte 32-Bit Borland Database Engine (BDE) zuständig. Neuartige Data Dictionaries und Datenmodule unterstützen den Entwickler auf jede denkbare Weise. Mit Datenintegrität und Geschäftsregeln ausgestattet, fühlen sich Ihre Anwendungen in einer dynamischen Geschäftsumgebung wie zu Hause.

In C++Builder geht jeder Datenzugriff mit den Komponenten der Seite *Datenzugriff* vor sich. Diese Objekte kapseln alle Informationen über die Datenquelle, beispielsweise die zugrundeliegende Datenbank, die Tabellen, auf die zugegriffen werden soll, die zu verwendende Abfrage oder bestimmte Feldreferenzen. Wenn Sie Anwendungen für DB2, Sybase, Microsoft SQL Server, Informix, Oracle, InterBase (einschließlich 95/NT-Lizenz für vier Benutzer), Paradox, dBASE und Local Interbase (gehört ebenfalls zum Lieferumfang) entwickeln, steht Ihnen die extrem leistungsfähige Borland Database Engine mit integrierten 32-Bit-Treibern zur Seite. Außerdem verfügt C++Builder über ODBC-Konnektivität für Server wie Access, VSAM oder die AS400. Der integrierte High-Speed-Zugriff auf Datenbankserver bedeutet höchste Effizienz für Ihre Client/Server-Anwendungen.

So erzeugen Sie eine Datenbankanwendung:

➔ **Wählen Sie im Hauptmenü *Datei|Neu***

➔ **Klicken Sie auf *Datenmodule***

➔ **Klicken Sie auf *Kundendaten* und bestätigen Sie mit *OK***

Die Datenmodule von *C++Builder*

Die Datenmodule von C++Builder Client/Server Suite bilden gewissermaßen den Informationspool Ihrer Anwendung, in dem Sie den Datenzugriff und die Geschäftsregeln zentral festlegen können. Datenmodule trennen Benutzeroberfläche und Geschäftsregeln und stellen eine Möglichkeit dar, solche Regeln ohne zusätzlichen Programmieraufwand zentral zu definieren und zu verwalten.

Sobald Sie ein Datenmodul erzeugt haben, ist es firmenweit verfügbar. Jedes Formular kann seine Anzeigeeigenschaften und Werte aus dem Datenmodul abrufen. Diese Trennung von Benutzeroberfläche und Daten bewirkt, daß Änderungen in einem der Bereiche keinerlei Auswirkungen auf den anderen haben. Jeder Bereich wird völlig eigenständig implementiert. Die Kenntnisse der Entwickler können ganz gezielt eingesetzt werden. Ein einzelner Programmierer benötigt keinen Doktorgrad in Datenbankdesign, Betriebswirtschaft **und** Oberflächenentwicklung mehr, um eine brauchbare Client/Server-Anwendung zu entwickeln.

Datenbankentwicklung per Drag&Drop

Borland C++Builder reduziert den Zeitaufwand für die Entwicklung von Datenbankanwendungen, indem viele sich wiederholende Vorgänge nur einmal durchgeführt zu werden brauchen. Die Wiederverwendung von Datenmodulen aus der Objektablage stellt eine Möglichkeit dar, eine Datenbankanwendung ohne jeden Aufwand mittels Drag&Drop zu erzeugen.

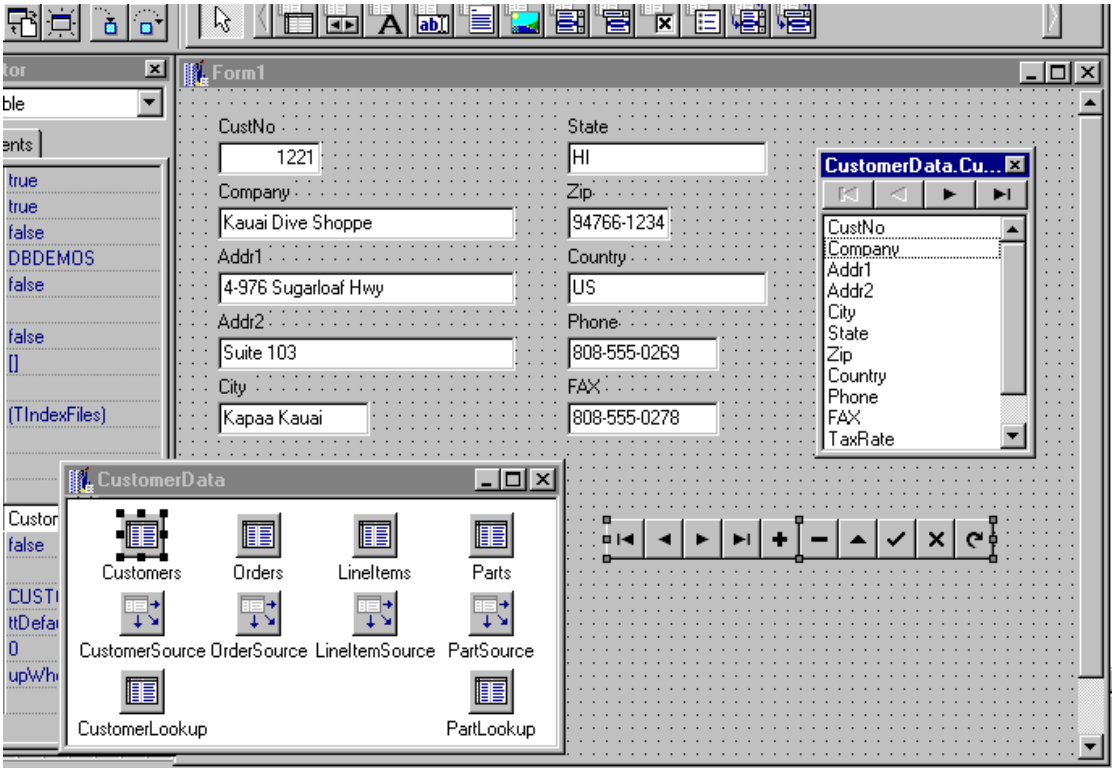
➔ **Doppelklicken Sie auf die Tabelle KUNDEN in Ihrem Datenmodul. Das Fenster *Customerdata.Customers* wird angezeigt.**

➔ **Markieren Sie mit der UMSCHALT-Taste und der Maus mehrere Felder.**

➔ **Ziehen Sie die markierten Felder auf das Formular.**

➔ **Fügen Sie einen Datenbanknavigator aus der Seite *Datensteuerung* hinzu und weisen Sie seiner Eigenschaft DataSource den Wert CustomerData->CustomerSource zu.**

Sie können sich nun im Entwurfsmodus mit Hilfe der Pfeile im Dialog *CustomerData.Customers* alle Datensätze der Tabelle anzeigen lassen. Compilieren Sie die Anwendung und führen Sie sie aus.



Einige der Datenbank-Tools von C++Builder: Datenmodule, Datenzugriffs- und Datensteuerungskomponenten

Zusammenfassung

Borland C++Builder für Windows 95 und NT, das bedeutet schnelle Entwicklung von Anwendungen per Drag&Drop, mehr als 100 wiederverwendbare Komponenten mit Quelltext und skalierbare Datenbank-Tools, kombiniert mit der Leistungsfähigkeit von C++. Ihre früheren Investitionen in C++ zahlen sich nun aus — C++Builder compiliert beliebigen ANSI C++ Code und unterstützt Sie bei der Entwicklung von schnellen und produktiven Front-Ends für Ihre Visual C++ und Borland C++ Anwendungen. Außerdem unterstützt C++Builder in vollem Umfang Industriestandards wie ANSI C++, Win32-API, ActiveX, OLE-Automation, ODBC, DCOM, MAPI, DirectX, Unicode, WinSock, ISAPI und NSAPI.

Neuere Versionen dieses Dokuments sind in der Borland Web Site unter www.Borland.com erhältlich.