Delphi für Windows

Moderne Windows-Entwicklungsumgebung mit Power-Features

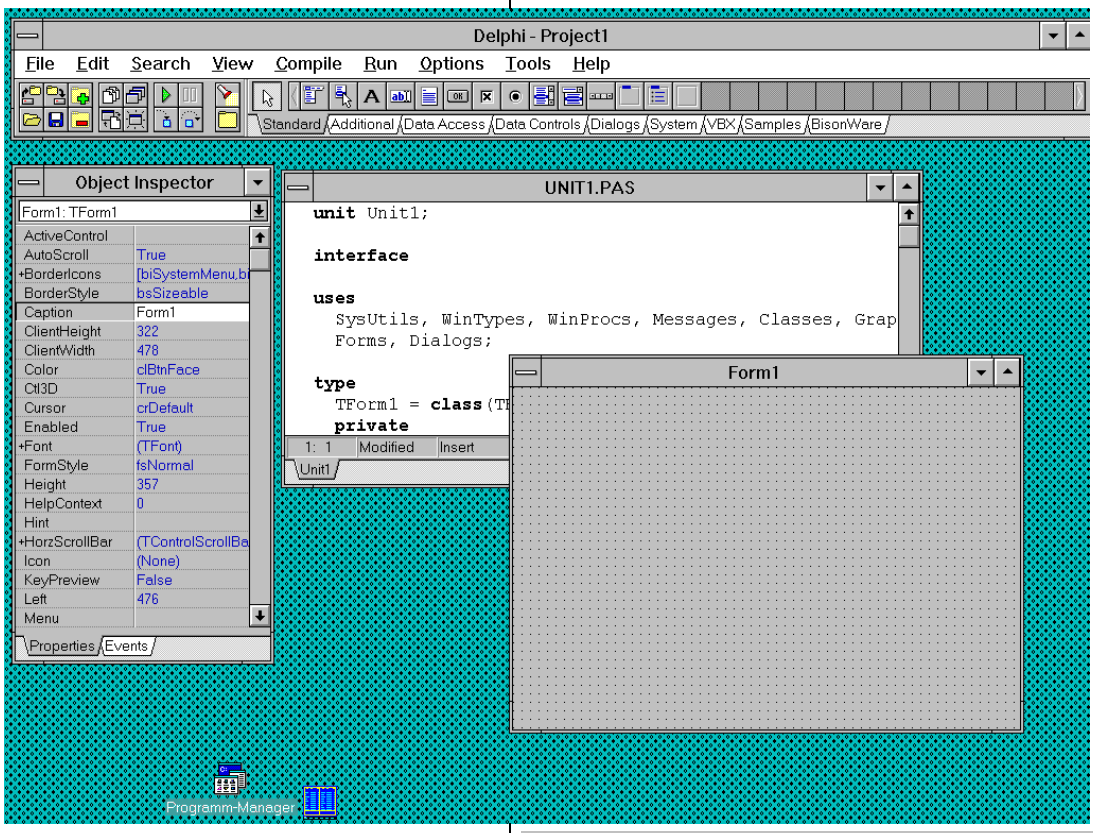
Andreas Zandomeneghi DSK 515:VB2DELPH.WRI

Einleitung

Seit etwa einem Jahr ist Delphi von Borland nun auf dem Markt und hat sich in der Zwischenzeit als modernes Entwicklungssystem unter Windows fest etabliert. Ich möchte daher heute die Gelegenheit ergreifen, Delphi etwas näher vorzustellen. Dieser Beitrag ist als Einführung gedacht. Eventuell werde ich in einer späteren Folge auf speziellere Detailaspekte eingehen.

Wie dem einen oder anderen Leser aus den Wirtschaftsteilen einschlägiger Tageszeitungen vielleicht bekannt ist, kämpfte die Firma Borland längere Zeit mit Problemen, v.a. deshalb, weil die meisten ihrer Produkte zu spät auf Windows umgestellt worden waren.

Delphi ist jedoch diesmal sicher das richtige Produkte zur richtigen Zeit: Während Windows-Entwicklungen auf C++ professionelle, geschwindigkeitsoptimierte Lösungen ergeben, aber sehr weitreichende, detaillierte Kenntnisse der Windows-Interna verlangen, ist Visual Basic zwar einfach zu bedienen, dafür sind die Programme wegen der Interpreter-Philosophie sehr langsam. Man kann also ohne weiteres behaupten, daß Delphi der erste große und erfolgreiche Wurf des Hauses Borland seit längerer Zeit darstellt.



Ich bitte dafür um Verständnis, daß diesem Artikel Delphi Version 1.0 zugrunde liegt. Die Version 2.0 gelangt in diesen Tagen in den Handel, unterscheidet sich in den Grundlagen aber nicht wesentlich von Delphi 1.0. Eventuell werde ich in einer der nächsten PCNEWS edit auf Änderungen und Verbesserungen eingehen.

Obwohl die Delphi-Ausführung als „Client Server Edition“ auf beinahe alle Arten SQL-Servern zugreifen kann, ist Delphi besser als Entwicklungsumgebung für allgemeine Applikationen zu bezeichnen, die außerdem zur Erstellung von Datenbank-Applikationen verwendet werden kann. Die Oberfläche ähnelt von der Bedienung her Visual Basic, obwohl die Delphi zugrunde liegende Programmiersprache Pascal und nicht Basic ist.

Da Delphi in englischer Sprache ausgeliefert wird, werde auch ich die originalen Bezeichnungen verwenden, bei Bedarf aber eine deutsche Übersetzung versuchen.

Die Delphi-Entwicklungsumgebung umfaßt eine Vielzahl von Komponenten (etwa 70), die jeweils für eine spezifische Aufgabe zugeschnitten sind. Es gibt neben den allgemeinen Window-Controls (Buttons, Edit-Boxen, Labels etc.) graphische Buttons, Notebook-Tabs (zum Blättern geeignete „Grafikseiten“), den Media-Player, OLE-Container und verschiedene Arten von Grids (Raster-Elemente).

Delphi unterstützt die aus Visual Basic bekannten VBX-Controls, allerdings nur solche der Version 1.0.

Delphi 1.0 generiert echten 16 Bit Code, und das Ergebnis ist eine einzige, direkt ablauffähige EXE-Datei, im Gegensatz zu Visual Basic, das interpreterorientiert arbeitet. Solche Programme sind bekanntlich ohne „Big Boss“ `VBRUN300.DLL` nicht lauffähig. Als Nachteil ist dagegen anzusehen, daß Delphi-Applikationen vergleichsweise groß sind; sie umfassen bereits für kleinere Projekte etwa 500kB. Nur für Datenbank-und Report-Funktionen und für wenige Sonderfälle werden in Delphi zusätzliche Runtime-DLLs und Zusatzdateien benötigt.

Bemerkenswert an dieser Entwicklungsumgebung ist noch, daß alle Komponenten selbst in Delphi geschrieben wurden.

Funktionen

Beim Start von Delphi sieht man vier Fenster:

1. Das Hauptfenster mit der Komponenten-Palette, Speedbar und Menu-Leiste
2. Den Object-Inspector
3. Den Source-Code Editor
4. Das Form-Window (leeres Formular)

Ich werde für die Besprechung mit einer weitergehenden Beschreibung der Funktionen beginnen, und dann anhand einfacher Beispiele die Arbeitsweise demonstrieren.

1. Menüleiste

Alle Befehle und Optionen sind über die Menü-Zeile erreichbar. Je nach Situation der aktivierten Fenster sind dabei nicht zur Verfügung stehenden Optionen eventuell grau ausgelegt.

Datei-Menü

Die hier zur Verfügung stehenden Optionen funktionieren so, wie die meisten Windows-Applikationen, z.B.:

- **New Project**: Erstellt neue Projekt-Datei (*.DPR). Die Projektdatei ist gewissermaßen die Datei, die alle Einzeldateien, aus denen eine Applikation besteht, zusammenhält, entsprechend der (*.MAK)-Datei in Visual Basic.
- **Save Project**: Speichert die Projekt-Datei und alle damit verbundenen Dateien.
- **Close Project**: Schließt aktuelles Projekt.
- **New Form**: Fügt neues, leeres Formular hinzu. (evt. mit Formularassistenten)
- **New Unit**: Fügt neue Unit (*.PAS) zum Projekt hinzu. Units sind nicht unbedingt mit einem Formular verknüpft, sondern können auch einzeln stehen.
- **New Component**: Erstellt eine neue Component (so bezeichnet man diesen Delphi-spezifischer Software-Baustein) mit Hilfe des „Komponenten-Experten".
- **Open File**: Öffnen einer einzelnen Datei.
- **Save Form**: Speichert die Datei, die gerade editiert wird.
- **Close File**: Schließt einzelne Datei, die momentan aktiv ist.
- **Add File**: Hinzufügen einer neuen Datei zum jeweiligen Projekt.
- **Remove File**: Löscht aktuelle Datei aus dem Projekt.

Edit-Menü

Hier stehen die aus Windows allgemein bekannten Befehle wie Cut, Paste etc. zur Verfügung. Außerdem können Controls ausgerichtet, vergrößert bzw. verkleinert werden, sowie die Taborder (Reihenfolge der Controls), wie aus VB bekannt, gesetzt werden.

Weitere Befehle:

- **Creation Order**: Zeigt bzw.verändert die Erstellungsreihenfolge von nicht grafischen Controls in einem speziellen Fenster
- **Lock Controls**: Controls „einfrieren", sodaß sie nicht mehr irrtümlich verschoben werden können, sowie das Wieder-Aufheben dieses Effektes.

Search-Menü

Dient zur Suche nach Text in der aktuellen Code-Datei. Es stehen u.a. Befehle wie „Find" und „Replace" und „Search Again" (F3) zur Verfügung.

View-Menü

Für größere Projekte erleichtert es dieses Menu, ein gewünschtes Fenster in der Anzeige nach vorne zu bringen.

- **Project Manager**: Mit Hilfe dieses Fenster können die einzelnen Code-und Formulardateien ausgewählt und angezeigt werden.
- **Object Inspector**: Zeigt den Object Inspector, der dem Eigenschafts-Fenster in Visual Basic entspricht.
- **Browser**: Zeigt Object Browser Fenster.

Außerdem können unter dem View-Menü verschiedene Hilfsfenster gewählt werden, die die Übersicht über gesetzte Breakpoints, den Stack, sowie verwendete Dateien und Forms, erleichtern.

Compile Menü

Einzelne Module, sowie das ganze Project können hier compiliert werden, sowie ein Syntax-Check durchgeführt werden. Ein Fenster mit Informationen über den Compiliervorgang steht außerdem zur Verfügung.

Run Menü

Enthält Optionen mit Run-Befehlen und zum Debuggen des Projekts. Die Debug-Optionen laufen nur, wenn unter dem „Project Options" Dialog „Include Debug Info" gewählt wurde. Die wichtigsten Optionen sind:

- **Run**: Build, startet dann die Applikation.
- **Parameters**: Eingabe von Run-Time Parametern
- **Step over**: Debugging: Springt über eine Codezeile.
- **Run To Cursor**: Die Funktionstaste F4 bewirkt, daß die Programmzeile in der der Cursor steht, rot unterlegt wird, und das Programm bei Durchlaufen dieser Position anhält.
- **Program Reset**: Ein Reset verursacht jedoch, daß Windows-Ressourcen evt. nicht wieder für Windows freigegeben wird, und so Speicher blockiert wird..
- **Add Breakpoint**: In einem Dialog-Fenster können Breakpoints gesetzt, editiert und gelöscht werden.
- **Evaluate/Modify**: Öffnet eine Dialog-Box, in der zu Debugging-Zwecken Variablen-Werte ausgelesen und geändert werden können.

Options Menü

Dieses dient zur Konfiguration der Entwicklungsumgebung. Folgendes sind die wichtigsten Optionen:

- **Project**: Enthält Optionen zur Einstellung der Formulare, für Programm-Icon, von Compiler, Linker, Verzeichnissen und für die Compiler-Direktiven.
- **Environment**: Optionen zur Konfiguration von Präferenzen, Editor, Library und Browser.
- **Tools**: Hier besteht die Möglichkeit, eigene Zusatzanwendungen und Hilfsprogramme aufzunehmen, die dann innerhalb Delphis unter „Tools" aufgerufen werden können.
- **Gallery**: Hier können die zur Verfügung stehenden „Gallerys", das sind vorbereitete Grundbausteine, für den Projekt- und den Formular-Experten eingestellt werden.
- **Install Components**: Es lassen sich neue Komponenten, sei es selbst entwickelte oder von Drittanbietern, installieren und auch wieder löschen. Auch VBX-Komponenten können hier aufgenommen werden.

Tools Menü

Enthält standardmäßig nur zwei Optionen:

- **Image Editor**: Zur komfortablen Erstellung von Icons und Bitmaps.
- **BDE Config**: BDE ist die Abkürzung für „Borland Database Engine". Es wird ein Hilfsprogramm zur Konfiguration für das BDE gestartet.

Help Menü

Neben den unter Windows allgemein bekannten Optionen steht die Option

- **Interactive Tutor** bereit, die den Neuling in verschiedene Teilbereiche von Delphi einführt.

Außerdem wird das Hilfsprogramm „Help Installer" mitgeliefert, um *.KWF (Keyword-File) Dateien zu definieren. Diese werden für kontextbezogene Hilfe gebraucht und greifen auf *.HLP Dateien zu.

Popup Menü

Wird die rechte Maustaste gedrückt, wenn der Mauscursor über einem der Delphi-Fenster steht, wird ein Popup-Menu mit häufig gebrauchten Befehlen (z.B. Configure, Help etc.) angezeigt.

2. Speedbar

Hier haben Sie Zugriff auf die am häufigsten gebrauchten Befehle, die oben bereits beschrieben wurden. Bei Rechtsklicken mit der Maus erscheint der Speedbar-Editor. Durch „Dragging" können neue Symbole daraus in die Speedbar aufgenommen werden.

3. Component Palette

Sie ist das zentrale Teil der Entwicklungsumgebung. Der Programmierer hat Zugriff auf etwa 70 verschiedene Komponenten, die in der Grundausstattung enthalten sind, außerdem auf evt. selbst erstellte und zugekaufte Komponenten. Die Properties (Eigenschaften) sind über den Object Inspector oder über Programmcode einstellbar.

Die Komponenten-Palette ist selbst als „Tabbed Component“ ausgebildet, ein Komponenten-Typ, der allgemein unter Delphi zur Verfügung steht. Jedem Tab ist eine Gruppe von Komponenten zugeordnet.

Wie im Fall der Speedbar ist auch hier eine individuelle Anpassung möglich. Selbstentwickelte Komponenten können eingefügt werden und danach wie solche von Delphi verwendet werden.

Komponenten können durch Ziehen mit der Maus oder durch Shift-Klick (mehrfach) auf das entsprechende Formular plaziert werden. Durch Doppelklick wird die Komponente automatisch zentriert.

4. Object Inspector

Nachdem die jeweilige Komponente auf dem Formular plaziert wurde, werden die Properties (Eigenschaften) mit dem Object Inspector gesetzt.

Der Object Inspector ist ebenfalls ein Tabbed Component, er enthält jedoch nur zwei Seiten: Properties (Eigenschaften) und Events (Ereignisse, das sind Aktionen, die einem solchen Element „passieren“ können- z.B. Mausklicks, Tastendrucke etc.).

Jede Seite ist in zwei Spalten unterteilt, links steht der Name des Properties, rechts der zugehörige Wert.

Am oberen Rand des Object Inspectors steht eine Liste von allen Controls, die in dem gerade aktuellen Formular enthalten sind. Es sind die Namen und der jeweilige Typ (z.B. `SpdBtnOpen: TSpeedbutton` bedeutetet, daß das Control mit Namen `SpdBtnOpen` vom Typ `Speedbutton` ist.) Wird ein Name in dieser Liste angewählt, selektiert Delphi automatisch das entsprechende Control im Formular.

Es können Properties und Events von mehr als einem Control dargestellt werden: Dazu <Shift>-klickt man auf alle gewünschten Objekte des Formulars. Im Object Inspector stehen dann nur die Eigenschaften die allen angewählten Controls gemeinsam sind.

Wechselt man mit dem Tab zu der Event-Seite des Inspectors, sieht man eine Auflistung aller Methoden, die das aktuelle Control besitzt. Wie von bei Visual bekannt, können selbstgeschriebene Funktionen bzw. Prozeduren dem gewünschten Ereignis zugeordnet werden. Durch Doppelklicken auf das rechte Feld/Events, gelangt man direkt in den Code-Editor, wo Delphi bereits automatisch ein Rahmengerüst für die Prozedur erstellt hat.

5. Forms

Wie unter Visual Basic bekannt, heißen die Fenster hier Forms (Formulare). Durch die Anwahl „New Form“ wird ein neues Formular erstellt. Dieses umfaßt zwei Dateien: Eine Datei mit Pascal-Quellcode (`*.PAS`), die mit „uses“ in das Projekt eingebunden wird, und eine binäre Datei, die den visuellen Aufbau des entsprechenden Formulars in Textform enthält (`*.DFM`). Lädt man eine DFM-Datei in Delphi, wird sie automatisch in Textform übersetzt und kann so direkt editiert werden.

Sehen wir uns beispielsweise das Form "UNIT1" an, das Delphi am Beginn eines neuen Projekts erstellt:

```
object Form1: TForm1
   Left = 200
   Top = 99
   Width = 435
   Height = 300
   Caption = 'Form1'
   Font.Color = clWindowText
   Font.Height
   Font.Name = 'System'
   Font.Style = []
   PixelsPerInch = 96
   TextHeight = 16
End
```

Es kann immer nur entweder die grafische Darstellung oder der Text angezeigt werden.

Ownership

Wird ein Control auf ein Formular plaziert, wird es „Besitzer“ dieser Elemente, das heißt, das Formular ist der Container für das Control. Es sind dabei auch verschachtelte Strukturen möglich, also kann ein Button als Container ein Panel (Rechteck-Flächen-Element) haben, das Panel wiederum hat als Container das Formular.

6. Der Code-Editor

Der Code Editor ist ebenfalls Beispiel der Zwei-Wege-Philosophie von Delphi, die wir eben gesehen haben. Plazieren wir zusätzliche Controls in einem Formular, wird automatisch neuer Quelltext in die PAS-Datei eingefügt. Dies können wir sehen, in dem wir das Code-Fenster neben der Formular anordnen. Fügen wir z.B. einen Button hinzu, wird in den Quelltext wie von Geisterhand

```
Button1: TButton;
```

eingefügt. Löschen wir den Button wieder, verschwindet diese Passage! Werfen wir einen näheren Blick auf den Quelltext, sehen wir, daß diese Veränderungen im „Type“-Bereich passieren.

```
type
  TForm1 = class(TForm)
  Button1: TButton;
  private
{ Private declarations }
  public
{ Public declarations }
end;
```

Wir sehen also, daß der Button das Objekt „Formular“ (vom Typ `TForm`) verändert. Das geänderte Formular erbt alle Eigenschaften des „Normalformulars“ (Höhe, Breite, Farbe etc.), es entsteht ein neuer, abgewandelter Typ `TForm1`, der Klasse `TForm`.

Der Code-Editor ist ebenfalls mit „Tabbed Pages“ wie die Komponenten-Palette und der Object-Inspector, ausgerüstet. Es kann somit bei größeren Applikationen zwischen den verschiedenen `*.PAS`-Dateien gewechselt werden.

Der Source-Code Editor ist übrigens an den populären „Brief“ Programmiereditor angelehnt, und mit Undo, Redo, Suchfunktionen sowie eine Fülle weiterer Möglichkeiten ausgestattet.

7. Projekt-Manager

Dieser ist ebenfalls formular-orientiert, und generiert zwei Dateien für jedes neue Formular der Applikation. Der Programm-Manager kann über das View-Menü erreicht werden, für zügigeres Arbeiten empfiehlt es sich jedoch, ihn direkt in der Speedbar zu installieren: Dazu bewegen Sie den Mauszeiger dorthin und Klicken die rechte Taste. Es erscheint ein kleines Pulldown-Menü, in dem Sie auf „Configure“ klicken. Der Speedbar-Editor ermöglicht nun, aus einer Liste „Commands“ auf der rechten Seite einzelne Icons auf die Speedbar zu ziehen, in diesem Fall also das „View Project Manager“-Icon.

8. Menu Designer

Von den verschiedenen Komponenten und ihren Möglichkeiten erscheint mir dieser besonders interessant. Wird die Komonente auf ein Formular plaziert, gelangt man durch Doppelklick in den Menü-Editor. Er ermöglicht mit geringem Aufwand, Windows-Menüs zu erstellen. Neue leere Menüpunkte werden einfach mit <INSERT> eingefügt, bzw. mit <DELETE> gelöscht. Es können außerdem „vorgefertigte" Menüs durch Rechtsklicken mit der Maus ("Template") eingebunden werden. Genauso besteht natürlich die Möglichkeit, eigene Menüs als Templates abzulegen.

9. Der Object Browser

Beim Compilieren des Projects werden detaillierte Informationen über alle Objekte, Units und globales Symbole abgelegt. Mit Hilfe des Object Browsers wird diese Information hierarchisch dargestellt. Information sind eher für „fortgeschrittene" Benutzer von Interesse.

10. Image Editor

Dieser ist ein eigenständiges Hilfsprogramm, mit dem auf einfache Art Bitmaps, Icons etc., erstellt werden können.

11. Environment Options

Dient zu persönlichen Konfiguration der Entwicklungsumgebung: Bildschirmfarben, Code-Editor Optionen etc. Auch für Compiler und Debugger kann hier ein „Fein-Tuning“ gemacht werden.

12. Experten

Für die Erstellung von neuen Formularen, Komponenten und Applikationen dienen diese dazu, diese Prozesse zu vereinfachen. Es wird dazu eine Abfolge von Fragen durchlaufen. Je nach den Wünschen des Anwenders wird das Objekt automatisch erstellt und kann dann weiter

„händisch“ angepaßt werden. Die Erstellung von eigenen Komponenten ist übrigens eher eine Sache für Fortgeschrittene; am Beginn findet man mit den vorinstallierten absolut das Auslangen.

13. Templates

Delphi nutzt die Möglichkeiten der objektorientieren Programmentwicklung sehr weitgehend aus. Dazu dient auch die Erstellung eigener, wiederverwendbarer Komponenten, sondern auch sogenannte Templates. Sie sind gewissermaßen ein Ausgangsrahmen für neue Projekte. Sie können für Projekte, Formulare und Menus erstellt werden.

Auch eigene Projekte können als Template abgelegt werden und erscheinen dann in einem Übersichtsfenster, der „Gallery“.

14. Gallery

Templates und Experten werden über die Gallery angewählt. Die Gallery wird übrigens bei Erstellung neuer Objekte nur dann angezeigt, wenn die Check-Boxen im Options Environment „Use on New Form“ und „Use on New Project“ gekreuzt sind.

15. Integrierter Debugger

Delphi wird mit einem integrierten Debugger geliefert. Auf Wunsch (Einstellung im Options-Menü) wird Debugging-Code generiert. Auf einfachsten Wege wird der Debugger aufgerufen, wenn man den Cursor auf eine Zeile setzt und F4 drückt (*Run to Cursor*). Das Programm wird gestartet und hält an, sobald der Code dieser Zeile erreicht wird.

Dokumentation

Leider muß über das Thema „Hilfe und Dokumentation“ hier auch einiges Negative gesagt werden: Es ist nämlich leider so, daß die Dokumentation nur für Anfänger ausreichend ist. Oft benötigt man aber detailliertere Informationen. In die Tiefe gehende Hilfestellung ist einfach nicht vorhanden, und auch Programmbeispiele sind eher dünn gesät. Wenn der Benutzer Delphis Objekt-Orientiertheit konsequent ausnützen möchte und nicht nur die häufigst verwendeten Methoden verwenden will, ist er auf Literatur bzw. die Online-Dienste angewiesen. Was Dokumentation betrifft, ist also Visual Basic Delphi (noch) einige Schritte voraus.

Eine Liste von empfehlenswerten Büchern folgte am Schluß dieses Artikels.

Meiner Meinung nach sollte man auf weiterführende Informationen nicht verzichten, um zu vermeiden, das Rad immer wieder neu zu erfinden.

Publikationen

Folgende monatlich erscheinende Magazine sind mir bekannt:

Delphi Informant

Erscheint bei Informant Communication in den USA. Das etwa 80-seitige Heft erscheint monatlich und erhält das Allerneueste von Delphi. Auch die Werbung ist interessant, da man hier zuerst von den allerneuesten Tools, Add-Ons etc. erfährt.

CompuServe: `GO ICGFORUM`
`http://www.informant.com`
Fax: 001-916-686-8497

Delphi Developer's Journal

Fax 001-502-491-8050

Delphi Developer

`http://www.pinub.com`
Fax 001-206-251-5057

Die beiden letztgenannten Magazine sind weniger umfangreich als der „Delphi Informant“ und enthalten keine Werbung. Technisch sind beide auf gutem, hohen Niveau.

Online-Unterstützung

Für Borland ist Compuserve der bevorzugte Online-Dienst. Folgende Foren werden angeboten:

`GO DELPHI`	Borland Delphi Forum (16Bit)
`GO BDELPHI32`	Borland Delphi 32 Forum (32Bit, d.h. Delphi 2.0)
`GO BORGER`	Borland Forum (deutsch)

Auch auf dem Internet sind natürlich eine Vielzahl von Anbietern, (kommerzielle und private „Cracks“) vertreten, die ich hier nicht alle aufführen kann. Am einfachsten, selbst auf dem Internet auf Entdeckungsreise gehen! (Ich verwende dazu übrigens den Online-Suchdienst „Lycos“, der in den letzten PCNEWS edit-Ausgabe sehr stiefmütterlich behandelt wurde...)

Das war also eine kurze Einleitung, der den Neuling mit den einfachsten Grundlagen von Delphi bekannt machen sollte.

In einer der nächsten Ausgaben werde ich mich mit der VCL, mit dem Thema Grafik und Delphi, sowie mit einfachen Programmbeispielen befassen.

Literaturliste

Delphi Developer's Guide (Xavier Pacheco/Steve Teixeira)
ISBN 0-672-30704-9

Delphi For Dummies (Neil Rubenking)
ISBN 1-56884-200-7

Delphi Unleashed
ISBN 0-67230499-6

Using Delphi (John Matcho/Eric Uber)
ISBN 1-56529823-3

Master Delphi (Charlie Calvert)
ISBN 0-672-30499-6

Visual Basic To Delphi
Write-Textdokument (siehe PCN-DSK-515)□


```
Apprentice Hacker
==================
   #!/usr/local/bin/perl
   $msg="Hello, world.\/n";
   if ($#ARGV >= 0)
   {
     while(defined($arg=shift(@ARGV)))
     {
       $outfilename = $arg;
       open(FILE, ">" . $outfilename) ||
            die "Can't write $arg:$!\/n";
       print (FILE $msg);
       close(FILE) || die "Can't close $arg: $!\/n";
     }
   }
   else
   {
     print ($msg);
   }
   1;
Experienced Hacker
==================
   #include <stdio.h>
   #define S "Hello, World\/n"
   main(){exit(printf(S) == strlen(S) ? 0 : 1);}


Seasoned Hacker
==================
   % cc -o a.out ~/src/misc/hw/hw.c
   % a.out


Guru Hacker
==================
   % cat
   Hello, world.
   ^D
```