Delphi

Einführung in Delphi

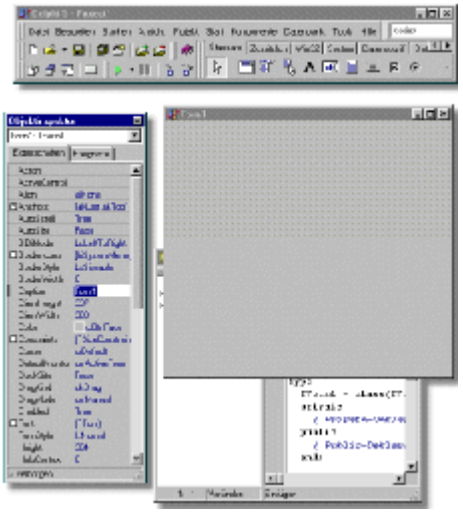
Bei Delphi handelt es sich um die visuelle Entwicklungsumgebung zu der Programmiersprache Pascal (genauer gesagt: zu Object Pascal).
Delphi ist das Konkurrenzprodukt zu Visual Basic im Bereich der "Nicht-Profi"-Sprachen (eine "Profi"-Sprache wäre z.B. C++, bzw. Visual C++). Im Internet herrscht eine rege Diskussion darüber welche Sprache "besser" ist. Ich enthalte mich der Diskussion und bemühe mich lieber euch die ersten Schritte mit Delphi beizubringen.
Wie auch Visual Basic bietet auch Delphi eine recht einfache Programmierung in einem Formular bei dem nur noch im Quellcodefenster die Ereignisse belegt werden müssen. Man sieht also, nicht wie bei Visual C++, was man programmiert. Es ist also im Grunde genommen eine WYSIWYG (What you see is what you get - man sieht vorher wie es später aussieht) - Programmiersprache.

Delphi im Detail

Delphi in der neuesten Version 8 gibt es in der Private Edition kostenlos, wer etwas mehr will bekommt dies mit der Pro Version, die aber immerhin 1.019 € kostet (Borland Delphi 8Pro *Stand 5/2004 Basis1*). Die kostenlose Variante kann man sich auf der Website von Borland herunterladen, natürlich nur wenn man eine ausreichende Bandbreite für den Download zur Verfügung und man sich dort angemeldet hat. Egal welche Version man sich besorgt, sie bietet wie Visual Basic und Visual C++ eine visuelle 32bit-Programmentwicklung für Windows.

Einführung

Die Programmierumgebung sieht etwas anders aus als bei Visual Basic oder Visual C++ - es ist schließlich kein Microsoft-Programm - denn die IDE (*integrated development environment*: Integrierte Entwicklungsumgebung = Programmierumgebung) besteht aus mehreren Fenstern, einem das die Symbol- und die Menüleiste enthält und die Steuerelemente beherbergt und einem Fenster, das die Eigenschaften des ausgewählten Objekts anzeigt, ein Formularfenster, das dem Formular in Visual Basic gleicht und zu guter Letzt das Quellcodefenster. Das Quellcodefenster ist im Gegensatz zu Visual Basic immer sichtbar, versteckt sich jedoch bei der Arbeit an dem Formular unter dem selbigen.



Das Formularfenster

Beginnen möchte ich mit dem zentralen, größten Fenster, dem Formularfenster in dem die Benutzeroberfläche des Programms entworfen, die Steuerelemente eingefügt und platziert werden.
Per Doppelklick auf ein Steuerelement gelangt man in das Quellcode-Fenster, das sich meistens hinter dem Formularfenster befindet. Hier können Anweisungen gegeben werden, was das Programm tun soll, wenn das Ereignis eintrifft, wie z.B. ein Klick auf einen Button.

Das Schaltpult

Ich nenne das obere Fenster mal das Schaltpult, da sich in diesem Fenster die Schalter befinden mit denen das Programm bedient wird. Neben den schon angesprochenen Steuerelementen befinden sich hier auch Schalter um das Programm zu starten, abzuspeichern und weitere grundlegende Einstellungen vorzunehmen. Außerdem beinhaltet es die Menüleiste, in der einige wichtige Menüpunkte zu finden sind.
Für jedes geladene Steuerelement gibt es in der Symbolleiste des Schaltpults ein kleines Symbol, diese Symbole sind in Rubriken eingeteilt. Klicke auf den Registerreiter um die Kategorie zu wechseln.
Will man das Steuerelement auf der Form platzieren, so klickt man das Symbol an und zieht es dann auf der Form auf. Also Symbol anklicken, mit Mauszeiger zur Form gehen, Maustaste klicken und gedrückt halten und ein Quadrat aufziehen, dann wieder los lassen und fertig ist das Steuerelement. Ist die Größe nicht perfekt oder ist die Position falsch, so kann man das Steuerelement auch noch verändern. Zum Verschieben auf das Steuerelement klicken, Maustaste gedrückt halten und an die korrekte Position verschieben. Zum Verkleinern Steuerelement anklicken, dann an der rechten unteren Ecke anklicken und bei gedrückter Maustaste in die richtige Größe verkleinern.

Das Eigenschaftenfenster

Das nächste Fenster ist das Eigenschaftenfenster, in dem sich sinnigerweise die Eigenschaften des aktuell ausgewählten Objektes befinden. Diese Eigenschaften können und müssen angepasst werden. Alle Steuerelemente haben Eigenschaften. Eine der Aufgaben des Programmierers ist es die anzupassen, so muss man manchmal die Beschriftung oder den Namen ändern. Dies alles ist im Eigenschaftendialog möglich.
Die Eigenschaften sind tabellarisch aufgelistet, neben dem Eigenschaftsname steht immer die aktuelle Einstellung dazu.
Um eine Eigenschaft zu ändern, klickt man auf die Einstellung daneben, daraufhin kann man bei manchen Eigenschaften in einer Liste die richtige Einstellung auswählen oder man muss die neue Eigenschafteneinstellung eingeben.
Will man die Eigenschaften eines anderen Steuerelements bearbeiten, so kann man dieses in der Liste über der Eigenschaftsanzeige.

Hilfe

Solltest du noch ein Element gefunden haben, das ich nicht beschrieben habe, dann verweile doch mit dem Mauszeiger über dem Element, dann wird dir in einem gelben Feld ein Tooltexttip gegeben. Es gibt auch eine Hilfe in der du bei Problemen nachsehen kannst.

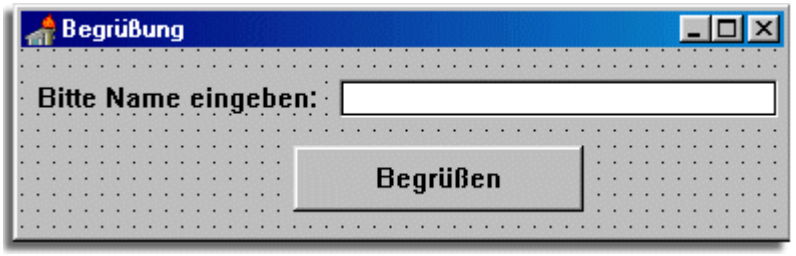
Weitergabe

Ausführbare Programme werden mittels des Menüpunktes "Projekt/*Projektname* erzeugen" erstellt, wobei Projektname für das gerade aktive Projekt steht. Nähere Ein-

stellmöglichkeiten zur ausführbaren Anwendung gibt es unter dem Menüpunkt "Projekt/Optionen".
Wenn du keine zusätzlichen Steuerelemente in dein Projekt integriert hast, dann ist die Datei auch auf anderen Computern ohne zusätzliche Dateien lauffähig. Die Ausnahme bildet hier die Datenbankprogrammierung, die noch eine zusätzliche Weitergabe von Datenbankdateien zum Programm erfordert, doch darauf will ich an dieser Stelle nicht weiter eingehen.

Delphi-Projekt anlegen

Als erstes wollen wir ein Hello World Programm programmieren. Dazu starte Delphi, dann sollte dich das Formularfenster begrüßen, in dieses fügen wir nun ein Label (A) in das Formularfenster ein. Diese Komponente wird ein Beschriftungsfeld. Bei der Label-Komponente verändern wir nun die "Caption"-Eigenschaft im Objektinspektor, mit ihr wird die Beschriftung der Label-Komponente verändert. Wir ändern sie in "Bitte Namen eingeben:". Daneben fügen wir nun die Edit-Komponente (ab|) ein. Sie soll den einzugebenden Namen aufnehmen. Damit beim Start des Programms die Edit-Komponente leer ist, muss die "Text"-Eigenschaft im Objektinspektor verändert werden. Diese Eigenschaft steuert den Inhalt der Komponente. Lösche also den Text aus der "Text"-Eigenschaft. Zum Schluss fügen wir noch die Button-Komponente (OK) ein. Sie soll später das Begrüßungsfenster auslösen. Die Beschriftung des Knopfes muss noch verändert werden, dies geschieht mit der mittlerweile schon bekannten Eigenschaft "Caption". Ändere sie in "Begrüßen" im Objektinspektor um. Nun sollte dein Formular in etwa so aussehen:



Quellcode für den Button eingeben

Nun geht es an das einfügen von Quellcode für die einzelnen Ereignisse (Sachen die passieren sollen). Doppelklicke auf die Button-Komponente und füge folgenden Quelltext zwischen die beiden Anweisungen *begin* und *end;* ein:

Name := Edit1.Text;
ShowMessage('Hello World. Hallo '+Name);

Bei *Name := Edit1.Text* wird der Variablen *Name* der Inhalt der Edit-Komponente zugewiesen (:= ist der Zuweisungsoperator). Die nächste Anweisung erzeugt ein Meldungsfenster mit der Aufschrift "Hello World. Hallo" und dann dem eingegebenen Namen, der in der Variable *Name* gespeichert ist. mit dem + vor dem Variablennamen wird die Variable mit dem Meldungstext verbunden (dies heißt mit Fachausdruck Zeichenkettenverkettung).

Variablen vereinbaren

Wenn wir das Programm jetzt ausführen wollten bekämen wir die Fehlermeldung "Typ nicht definiert" und das mit gutem Grund! Wir müssen erst die Variable vereinbaren. Scrolle in dem Quelltextfenster weiter hoch, bis du die Anweisung *var Form1:*

TForm1; findest, füge dahinter die Anweisung
Name: String;

an. Hier mit wird die Variable *Name* vereinbart. Die Vereinbarung funktioniert folgendermaßen:
Variablenname: Variablentyp;.

Nun dürfte unser erstes Programm laufen. Zum Testen drücke [F9], nun sollte das Programm starten. Wenn ihr euren Namen eingegeben habt und auf den "Begrüßen"-Button gedrückt habt, gegrüßt das Programm die Welt und euch.

Was beim Speichern zu beachten ist

Noch was zum Speichern: Wenn du abspeicherst, achte darauf, dass die .pas-Datei und die Projektdatei (.dpr) unterschiedliche Namen haben!

Noch ein Projekt

Ich möchte noch ein anspruchsvolleres Programm schreiben, da das Begrüßungsprogramm doch sehr einfach war. Ich habe vor, ein Zahlenratespiel zu programmieren.

Formulardesign

Lege ein neues Projekt an, ziehe nun eine Label-Komponente auf und platziere sie im oberen Drittel des Formulars. Ändere dann die Eigenschaft "Caption" in "Ich denke mir eine Zahl zwischen 1 und 1000". Erstelle dann eine Button-Komponente und ändere die "Caption"-Eigenschaft in "Rate mal" um. Nun sollte dein Formular so aussehen:

Quellcode für den Start des Formulars eingeben

Nun ist die Arbeit im Formular abgeschlossen. Nun geht es an die Arbeit im Quelltext. Es soll jedes Mal eine neues Zahl erfunden werden, dazu wird eine Zufallszahl erstellt. Da die Zufallszahl schon zum Start des Programms verfügbar sein muss, werden die Anweisungen dazu in die FormCreate-Prozedur geschrieben. Doppelklicke dazu auf eine freie Fläche im Formular. Gib dann folgenden Quelltext zwischen die beiden Anweisungen *begin* und *end;* ein:

```
Randomize;
Zufall := Random (1000) + 1;
Zahl := 500;
```

Mit der Anweisung *Randomize* wird der Zufallszahleneditor gestartet, danach wird der Variable *Zufall* die Zufallszahl zugewiesen. Dies geschieht folgendermaßen: ***Random(Zahlenraum)***, d.h. eine Zufallszahl im Zahlenraum bis 1000 soll erstellt werden. Nun ist da aber ein Denkfehler vorhanden: Delphi fängt mit 0 an zu zählen so dass auf die Zufallszahl noch eins dazu gezählt werden muss, bis die gewünschten 1 bis 1000 vorhanden sind. Die letzte Anweisung weist der Variable *Zahl* den Wert

500 zu, du wirst später erfahren warum das nötig ist.

Variablen vereinbaren

Damit wir es später nicht vergessen müssen wir noch die Variablen *Zufall* und *Zahl*-vereinbaren. Suche also wieder die Anweisung *var* [...] und füge folgendes dahinter an:

Zufall, Zahl: Integer;

Hier siehst du gleich, dass man mehrere Variablen auf einmal vereinbaren kann.

Quellcode für den Button eingeben

Nun wollen wir mal die Anweisungen für den Knopf einfügen und das sind einige. Doppelklicke also auf die Button-Komponente und gib folgendes zwischen die beiden Anweisungen *begin* und *end;* ein:

```
Eingabe := InputBox('Rate mal', 'Zahl eintippen', IntToStr (Zahl));
Zahl := StrToInt (Eingabe);

{Aktuelle Eingabe auswerten, ob zu klein / groß}
If Zahl < Zufall Then
Begin
Label1.Caption := 'Deine Zahl ist zu klein!';
Vers := Vers + 1;
End;
If Zahl> Zufall Then
Begin
Label1.Caption := 'Deine Zahl ist zu groß!';
Vers := Vers + 1;
End;

{Wenn richtig geraten, neues Spiel anbieten}
If Zahl = Zufall Then
Begin
Label1.Caption := 'RICHTIG!!!';
ShowMessage ('Du hast ' +IntToStr (Vers)+ 'mal geraten');
Knopf := Application.MessageBox ('', 'Neues Spiel?', 32+4);

{Wenn neues Spiel gewünscht, Startwert/Zufallszahl neu}
If Knopf = idYes Then
Begin
Label1.Caption := 'Ich denke mir eine Zahl!';
Button1.Caption := 'Rate mal!';
Zufall := Random (1000) + 1;
Zahl := 500;
Vers := 0;
End

{Wenn kein neues Spiel, Programmende}
Else
Close;
end
```

{Wenn nicht richtig geraten, noch mal}
Else
Button1.Caption := 'Noch mal!';

Mit der ersten Anweisung wird der Variable *Eingabe* der Inhalt der *InputBox* zugewiesen. Die *InputBox* ist ein Eingabefenster in das die geratene Zahl kommen soll. Wobei der erste Textabschnitt ('Rate mal') der Titel ist, der nächste Textabschnitt der Text und der letzte der Vorgabewert der im Eingabefeld steht. Hier kommt beim erste Aufruf die Zahl 500 herein die wir in der FormCreate-Prozedur der Variable *Zahl* zugewiesen haben. *IntToStr* ist ein sogenannter Umwandlungsoperator, du musst wissen, dass wir die Variable *Zahl* als *Integer* vereinbart haben, eine *InputBox* aber nur *String*-Zeichenketten aufnehmen kann. So wandeln wir einfach die *Integer*-Zahl in eine *String*-Zeichenkette um. Die Anweisung darunter macht genau das Gegenteil, sie wandelt die eingegebene *String*-Zeichenkette in eine *Integer*-Zahl um, damit wir sie mit der Zufallszahl vergleichen können.

Bedingungsabfrage

Bei den nächsten Zeilen handelt es sich um eine Bedingungsabfrage, mit ihnen wird überprüft ob eine bestimmte Bedingung erfüllt ist - bei uns ob die eingegebene Zahl mit der Zufallszahl übereinstimmt. Die Struktur sieht folgendermaßen aus: *If Bedingung Then Anweisung*. Da bei uns mehrere Anweisungen ausgeführt werden müssen, müssen die Anweisungen in *begin* und *end;* gesetzt werden. So sieht die Struktur folgendermaßen aus:
If Bedingung Then
Begin
Anweisung
End;

Die ersten beiden If-Strukturen überprüfen, ob die eingegebene Zahl größer oder kleiner als die Zufallszahl ist, und ändern die Beschriftung der Label-Komponente entsprechend. Dann wird die Variable *Vers* um eins erhöht, was es mit dieser Variable auf sich hat werde ich in Kürze auflösen.

Verschachtelte Bedingungsabfrage

Die dritte Bedingungsabfrage ist für den Fall das man richtig geraten hat, sie ist eine verschachtelte If-Then-Else-Konstruktion da bei Erfüllung der Bedingung wieder eine Bedingung abgefragt wird. Wenn die erste Bedingungen nicht erfüllt wird, wird mittels *else* eine Alternative ausgeführt. Dann wird die Beschriftung der Label-Komponente in "Richtig!!!" geändert.

Die Variable *Vers*

Nun kommt unsere Variable *Vers* zum Einsatz. Mit ihr wird angegeben wie oft geraten wurde, da es sich um eine *Integer*-Variable handelt wird, muss diese mittels *IntToStr* in eine Zeichenkette umgewandelt werden. Den Zeichenkettenverkettungsoperator (+) kennst du ja bereits.

Die anderen Anweisungen

Mit der nächsten Anweisung wird der Variable *Knopf* eine *MessageBox* (ist in etwa das gleiche wie eine *ShowMessage*-Anweisung, man kann jedoch mehr Parameter einstellen) zugewiesen. Die Meldung hat keine Beschriftung, der Titel ist "Neues

Spiel?" und die Zahlen geben an wie die Meldung aussehen soll (32 steht für Fragezeichen (16 für Warnkreuz, 48 für Ausrufezeichen und 64 für Infozeichen) und 4 steht für die Knöpfe 'Ja' und 'Nein' (0 für OK, 1 für OK und Abbrechen, 2 für Abbrechen und Wiederholen und Ignorieren, 3 für Ja und Nein und Abbrechen und 5 steht für Wiederholen und Abbrechen).
Die nächste Anweisung überprüft ob 'Ja' gedrückt wurde (idYes), falls ja, dann wird ein neues Spiel gestartet und dazu die einzelnen Variablen auf ihren Standartwert zurück gebracht. Wenn nicht 'Ja' gedrückt wurde, also 'Nein', dann wird das Spiel beendet. Wenn falsch geraten wurde, dann wird die Beschriftung des Buttons in "Noch mal" geändert.

Variablen vereinbaren

Nun müssen nur noch die Variablen *Eingabe*, *Knopf* und *Vers* vereinbart werden. Erweitere die Integervereinbarung so weit, dass sie folgendermaßen aussieht:
Zufall, Zahl, Knopf, Vers: Integer;
und schreibe dahinter noch:
Eingabe: String;.
Nun sollte das Zahlenratespiel funktionieren.

Zur Weitergabe

Zur Weitergabe der selbsterstellten Programme ist nur zu sagen, dass sie ohne zusätzliche Dateien laufen. Also brauchst du keine zusätzliche Systemdateien mitzuliefern und ein Installationsprogramm erzeugen.