最新程序开发软件集锦

Delphi 4.0 开发程序集

窦万峰 张子瑜 王保保 编著

东南大学出版社

Delphi 4.0

TP312
DWF/1

Delphi4.0 开发程序集

窦万峰　张子瑜　王保保　**编著**

东南大学出版社

内 容 提 要

Delphi 可视化开发工具已经受到计算机编程人员的喜爱和广泛使用。随着对 Delphi 的学习，广大用户希望有一本关于 Delphi4.0 实用程序集的书，以便能够快速和深入地掌握 Delphi4.0 编程的技巧和本质。

本书分为六大部分，包括 Delphi4.0 基本程序开发、菜单与多窗体及文件操作编程、图形图像应用、多媒体编程、数据库应用和其他实用编程知识。全书汇编了 80 多个开发实例，重点介绍了程序开发方面的技巧。本书的例程由简单到复杂，并稍作修改就能直接应用到用户的程序中。

本书既适合于初学者，又可为程序开发人员使用，是 Delphi4.0 编程的进一步深入。

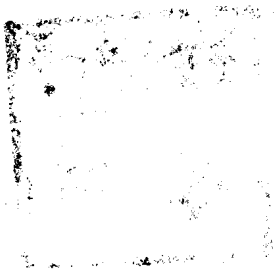
JS371/18

图书在版编目(CIP)数据

Delphi 4.0 开发程序集/窦万峰等编著.-南京:东南大学出版社,
1999.8
ISBN 7-81050-510-6

Ⅰ.D... Ⅱ.窦... Ⅲ.DELPHI 语言-程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字 (1999)第 32451 号



东南大学出版社出版发行
(南京四牌楼 2 号 邮编 210096)
出版人：洪焕兴
江苏省新华书店经销 江浦第二印刷厂印刷
开本：787mm×1092mm 1/16 印张:21.5 字数:537 千字
1999 年 8 月第 1 版 1999 年 8 月第 1 次印刷
印数:1-4000 册 定价:28 元

前　言

Delphi 软件开发工具是 Borland 公司推出的一个完全导向的可视化系统开发环境的工具，被评为美国最优秀软件之一。Delphi 结合了可视化技术、面向对象技术、数据库技术等多种先进的软件编程技术和思想，并使用了世界上最快的编译器，使其成为创建功能丰富、界面友好的 Windows 应用软件工具之一。

1995 年 7 月，Borland 公司推出 Delphi1.0，它所具有的丰富功能引起人们的关注。一年后推出的 Delphi2.0，引起广大计算机用户的青睐，其强大功能更使人们欢欣鼓舞。Delphi3.0 于 1997 年 5 月正式推出，其优良、稳定的性能，强大的数据库支持，更快的速度以及全面支持 Internet 等等，更成为万众瞩目的焦点。Delphi4.0 加强了数据库与网络程序开发的功能，并增加了许多最新的技术。

作者在使用 Delphi 开发应用程序经验的基础上，结合一些最新资料和大量开发实例，汇编了这本 Delphi4.0 程序集。本书的许多例子都是从我们开发的程序和其他实际应用程序中摘选来的，因而更具实用性。我们衷心希望这些例程能够对读者学习 Delphi 和程序开发带来有益的帮助。

本书的内容共分为 6 章：

第 1 章介绍了 Delphi4.0 的基本编程实例，包括基本窗体制作、组件的使用以及代码的编写等内容。

第 2 章通过大量实例介绍菜单制作技术、多窗体应用编程、文件对话框应用程序开发以及文件操作应用。

第 3 章主要给出与图形图像处理有关的程序和动画应用程序以及图表制作。包括画布对象应用程序、鼠标绘图程序、图形图像输出程序、动态调用程序，以及各种图表数据加入和动态控制等。

第 4 章主要介绍多媒体应用编程，包括播放声音文件程序、电视伴音程序以及电影动画制作程序等。

第 5 章主要涉及 Delphi4.0 的数据库应用程序，包括数据显示程序、数据输入程序、数据查询和字段处理程序、主细表应用程序和多线程数据库应用开发等。

第 6 章介绍了 Delphi4.0 其他应用程序，包括超文本帮助文件制作和应用程序开发、组件制作和调用程序、外部执行文件调用以及异常处理程序等。

本书中列举了大量的编程实例，循序渐进地介绍了从 Delphi4.0 基本程序编写到图形图像数据库及多媒体应用程序开发，使读者能深入地掌握 Delphi4.0 编程技巧及其实用开发技术。

本书编写过程中得到了刘菁华同志的大力支持和帮助，在此表示衷心感谢。

本书适用于广大计算软件开发人员和计算机爱好者学习和使用。

由于时间和水平有限，不妥之处在所难免，敬请广大读者斧正，不胜感激。

编　者

1999.7

目 录

此为试读,需要完整PDF请访问: www.ertongbook.com

第一章　Delphi4.0 基本编程实例

本章通过大量的实例介绍使用 Delphi4.0 编程的基本方法和过程，以及窗体制作和组件使用等。通过这些基本实用的例子，读者会快速掌握 Delphi4.0 编程的过程和它的实质，领略到 Delphi4.0 的强大功能和一些编程技巧。读者也可以了解到使用 Delphi4.0 能够开发具有各种功能的 Windows 应用程序，而且使用起来非常得心应手。

1.1 数据类型和类方法编程举例

本节通过几个实例介绍如何使用 Delphi 定义数据类型、类和类方法，以及如何操作使用这些类和方法等。

1.1.1 数据类型测试程序

下面是一个数据类型测试程序，它可以测试各种类型的数据所占用的字节大小、所能取的最大值和最小值。程序窗体如图 1-1 所示。

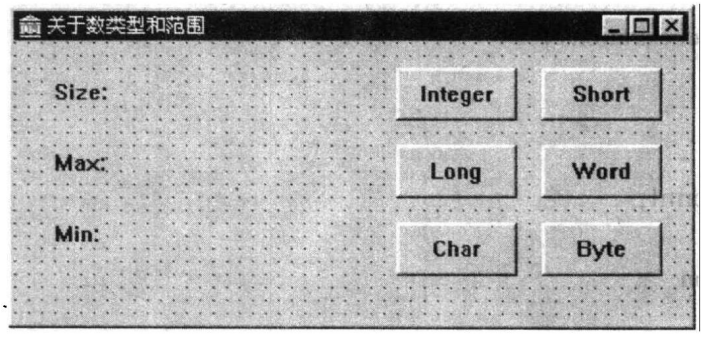


图 1-1　数据类型测试程序窗体

数据类型测试程序代码如下：

```
unit Test;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    SizeLabel: TLabel;
```

```
    MaxLabel: TLabel;
    MinLabel: TLabel;
    IntButton: TButton;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    ShortButton: TButton;
    LongButton: TButton;
    WordButton: TButton;
    CharButton: TButton;
    ByteButton: TButton;
    procedure IntButtonClick(Sender: TObject);
    procedure ShortButtonClick(Sender: TObject);
    procedure LongButtonClick(Sender: TObject);
    procedure WordButtonClick(Sender: TObject);
    procedure CharButtonClick(Sender: TObject);
    procedure ByteButtonClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}
{测试整数类型的字节大小、最大值和最小值}
procedure TForm1.IntButtonClick(Sender: TObject);
var
   Number: Integer;
begin
  Number := 34;
  SizeLabel.Caption := IntToStr (SizeOf (Number));
  MaxLabel.Caption := IntToStr (High (Number));
  MinLabel.Caption := IntToStr (Low (Number));
end;
{测试短整数类型的字节大小、最大值和最小值}
procedure TForm1.ShortButtonClick(Sender: TObject);
var
   Number: ShortInt;
```

```
begin
   Number := 34;
   SizeLabel.Caption := IntToStr (SizeOf (Number));
   MaxLabel.Caption := IntToStr (High (Number));
   MinLabel.Caption := IntToStr (Low (Number));
end;
{测试长整数类型的字节大小、最大值和最小值}
procedure TForm1.LongButtonClick(Sender: TObject);
var
    Number: LongInt;
begin
   Number := 34;
   SizeLabel.Caption := IntToStr (SizeOf (Number));
   MaxLabel.Caption := IntToStr (High (Number));
   MinLabel.Caption := IntToStr (Low (Number));
end;
{测试字类型的字节大小、最大值和最小值}
procedure TForm1.WordButtonClick(Sender: TObject);
var
    Number: Word;
begin
   Number := 34;
   SizeLabel.Caption := IntToStr (SizeOf (Number));
   MaxLabel.Caption := IntToStr (High (Number));
   MinLabel.Caption := IntToStr (Low (Number));
end;
{测试字符类型的字节大小、最大值和最小值}
procedure TForm1.CharButtonClick(Sender: TObject);
var
    Number: Char;
begin
   Number := 'x';
   SizeLabel.Caption := IntToStr (SizeOf (Number));
   MaxLabel.Caption := IntToStr (Ord (High (Number)));
   MinLabel.Caption := IntToStr (Ord (Low (Number)));
end;
{测试字节类型的字节大小、最大值和最小值}
procedure TForm1.ByteButtonClick(Sender: TObject);
var
    Number: Byte;
begin
   Number := 34;
```

```
  SizeLabel.Caption := IntToStr (SizeOf (Number));
  MaxLabel.Caption := IntToStr (High (Number));
  MinLabel.Caption := IntToStr (Low (Number));
end;

end.  {结束}
```

下面的例子介绍数据类型的转换过程，这个程序比较简单，我们直接给出其代码：

```
unit number;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls;

type
  TNumbersForm = class(TForm)
    Label1: TLabel;
    Edit1: TEdit;
    Label2: TLabel;
    Edit2: TEdit;
    Label3: TLabel;
    Edit3: TEdit;
    Label5: TLabel;
    Edit5: TEdit;
    Label4: TLabel;
    Edit4: TEdit;
    CheckButton: TButton;
    procedure CheckButtonClick(Sender: TObject);
    procedure Edit2Exit(Sender: TObject);
    procedure Edit3Change(Sender: TObject);
    procedure Edit4KeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  NumbersForm: TNumbersForm;

implementation
```


此为试读，需要完整PDF请访问：www.ertongbook.com

```
{$R *.DFM}
{下面的过程测试 Edit1 编辑框中的字符是否为普通字符}
procedure TNumbersForm.CheckButtonClick(Sender: TObject);
var
  Number, Code: Integer;
begin
  if Edit1.Text <> '' then
  begin
    val (Edit1.Text, Number, Code);
    {如果字符转换数字不成功}
    if Code <> 0 then
    begin
      Edit1.SetFocus;  {使 Edit1 仍处于聚焦状态}
      MessageDlg ('Not a number in the first edit', mtError, [mbOK], 0);
    end
    else  {若成功，则显示成功信息}
      MessageDlg ('OK, in the first edit there is a number', mtInformation, [mbOK], 0);
  end;
end;
{Edit2Exit 过程将字符转换成数字，其在焦点离开 Edit2 编辑框时发生}
procedure TNumbersForm.Edit2Exit(Sender: TObject);
var
  Number, Code: Integer;
begin
  if (Sender as TEdit).Text <> '' then
  begin
    val ((Sender as TEdit).Text, Number, Code);
    if Code <> 0 then
    begin
      (Sender as TEdit).SetFocus;
      MessageDlg ('The edit field number ' +IntToStr ((Sender as TEdit).Tag) +
        ' does not have a valid number', mtError, [mbOK], 0);
    end
  end;
end;
{下面的过程将字符转换成数字}
procedure TNumbersForm.Edit3Change(Sender: TObject);
var
  Number, Code: Integer;
begin
  if (Sender as TEdit).Text <> '' then
  begin
```

```
    val ((Sender as TEdit).Text, Number, Code);  {字符转换}
    if Code <> 0 then  {若不成功，则显示错误信息}
      MessageDlg ('The edit field number ' +IntToStr ((Sender as TEdit).Tag) +
        ' does not have a valid number', mtError, [mbOK], 0);
  end;
end;
{下面的过程测试键入的值是否为数字}
procedure TNumbersForm.Edit4KeyPress(Sender: TObject; var Key: Char);
begin
  {检查键入的值是数字还是非数字值}
  if not (key in ['0'..'9', #8]) then
  begin
    Key := #0;
    MessageBeep ($FFFF);  {非数字则发出鸣叫}
  end;
end;

end.  {结束}
```

1.1.2 字符排序程序

本例使用选择排序法，即将扫描所有数据并找出最小值，然后将其放在第一个位置，接着扫描剩余的数据项，找出下一个最小值，依次类推，直到最后一个数据项。

建立一个新窗体，放入一个 List 组合框，将其 Name 属性设为 Listarray，将其 Caption 属性设为 List，并保存这个项目为 ProSort.dpr，单元为 Sort，窗体的标题为 TestForm。

在窗体中加入两个按钮，一个标题为 Create，用于创建随机数据；另一个的标题为 Sort，用于排序。

首先声明一个数组变量：

```
Var
  Testsort:TTestSort;
  Data: Array[1..30] of char;
```

双击 Create 按钮激活 CreateClick 过程，并加入如下代码：

```
procedure TTestSort.CreateClick(Sender: TObject);
var
  i:integer;
begin
  List.caption:='';
 for i:=1 to 30 do
 begin
  Data[i]:=chr(Random(26+65));
```

```
        List.caption:=List.caption+Data[i]+',';
        update;
      end;
    end;
```

用 Update 过程更新列表框的数据显示。由于字母 A 到 Z 的 ASCII 码值为 65 到 90，故使用下列代码产生一个随机字符：

```
    Data[i]:=chr(Random(26+65));
```

双击 Sort 按钮，在 SortClick 过程中加入如下的代码：

```
  procedure TTestSort.SortClick(Sender: TObject);
  var
   i,j,low,index:integer;
   Hold:char;
  begin
    for i:=1 to 29 do
    begin
      low:=i;
      for j:=i+1 to 30 do
       if Data[j]<Data[low] then low:=j;
        if i<>low then
        begin
          Hold:=Data[i];
          Data[i]:=Data[low];
          Data[low]:=Hold;
          List.caption:='';
          for index:=1 to 30 do
          begin
            List.caption:=List.caption+Data[index]+',';
            update;
          end;
        end;
     end;
    end;
  end.
```

程序中使用了一个循环更新语句，用于每做一次排序都要对列表框进行重新显示，因而用户可动态地观察排序过程。

运行该程序，其结果如图 1-2 所示。

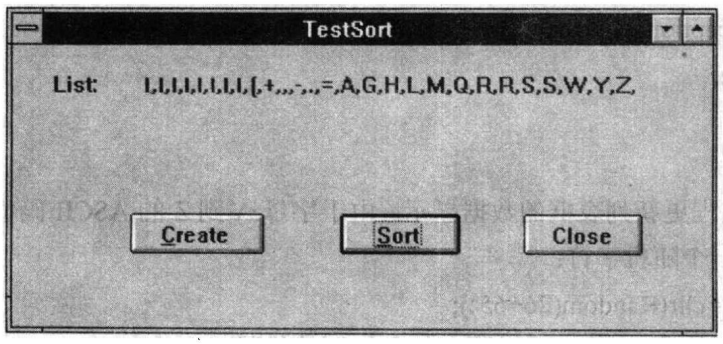


图 1-2　字符排序演示

1.1.3　造句程序

造句程序指将不同组的单词连接成一个完整的句子。造句程序窗体如图 1-3 所示。

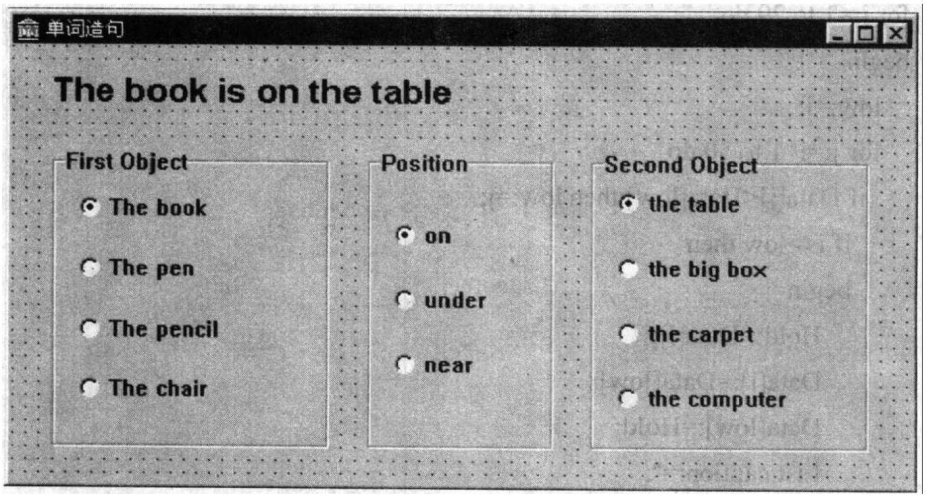


图 1-3　造句程序窗体

造句程序代码如下：

```
unit phaselink;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    GroupBox1: TGroupBox;
```

```
    GroupBox2: TGroupBox;
    GroupBox3: TGroupBox;
    RadioBook: TRadioButton;
    RadioPen: TRadioButton;
    RadioPencil: TRadioButton;
    RadioOn: TRadioButton;
    RadioUnder: TRadioButton;
    RadioNear: TRadioButton;
    RadioChair: TRadioButton;
    RadioTable: TRadioButton;
    RadioBox: TRadioButton;
    RadioCarpet: TRadioButton;
    RadioComputer: TRadioButton;
    Label1: TLabel;
    procedure ChangeText(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}
{ ChangeText 过程完成字符的连接}
procedure TForm1.ChangeText(Sender: TObject);
var
  Phrase: String;
  I: Integer;
begin
  Phrase := '';
  {确定选择哪一个 radiobutton，然后将它的文本加到句子中}
  for I:=0 to GroupBox1.ControlCount - 1 do
    if (GroupBox1.Controls[I] as TRadioButton).Checked then
      Phrase := Phrase + (GroupBox1.Controls[I] as TRadioButton).Caption;
  {加入动词 is 和一个空格}
  Phrase := Phrase + ' is ';
  {同样重复进行 groupbox2 中的单词}
  for I:=0 to GroupBox2.ControlCount - 1 do
```

```
        if (GroupBox2.Controls[I] as TRadioButton).Checked then
          Phrase := Phrase + (GroupBox2.Controls[I] as TRadioButton).Caption;
      {加入一个空格}
      Phrase := Phrase + ' ';
      {处理 groupbox3 中的单词}
      for I:=0 to GroupBox3.ControlCount - 1 do
        if (GroupBox3.Controls[I] as TRadioButton).Checked then
          Phrase := Phrase + (GroupBox3.Controls[I] as TRadioButton).Caption;
      {显示整个句子}
      Label1.Caption := Phrase;
    end;
    end.{结束}
```

下面这个程序完成另一类句子的造句过程，而且用户还可以加入新的单词。程序窗体如图 1-4 所示。

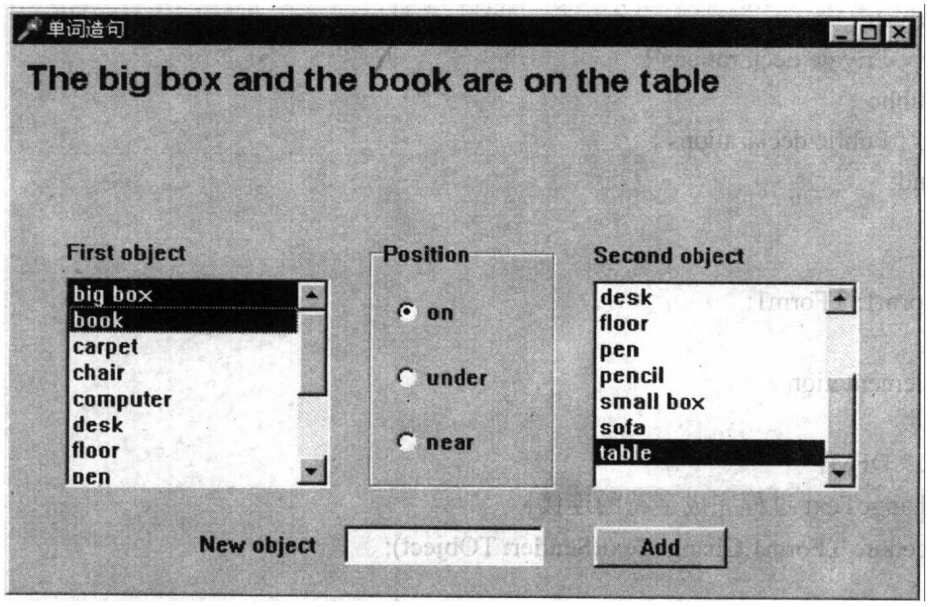


图 1-4　另一类造句程序窗体

程序代码如下：

```
unit phaselink2;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes,
  Graphics, Controls, Forms, Dialogs, StdCtrls;
```


此为试读，需要完整PDF请访问：www.ertongbook.com