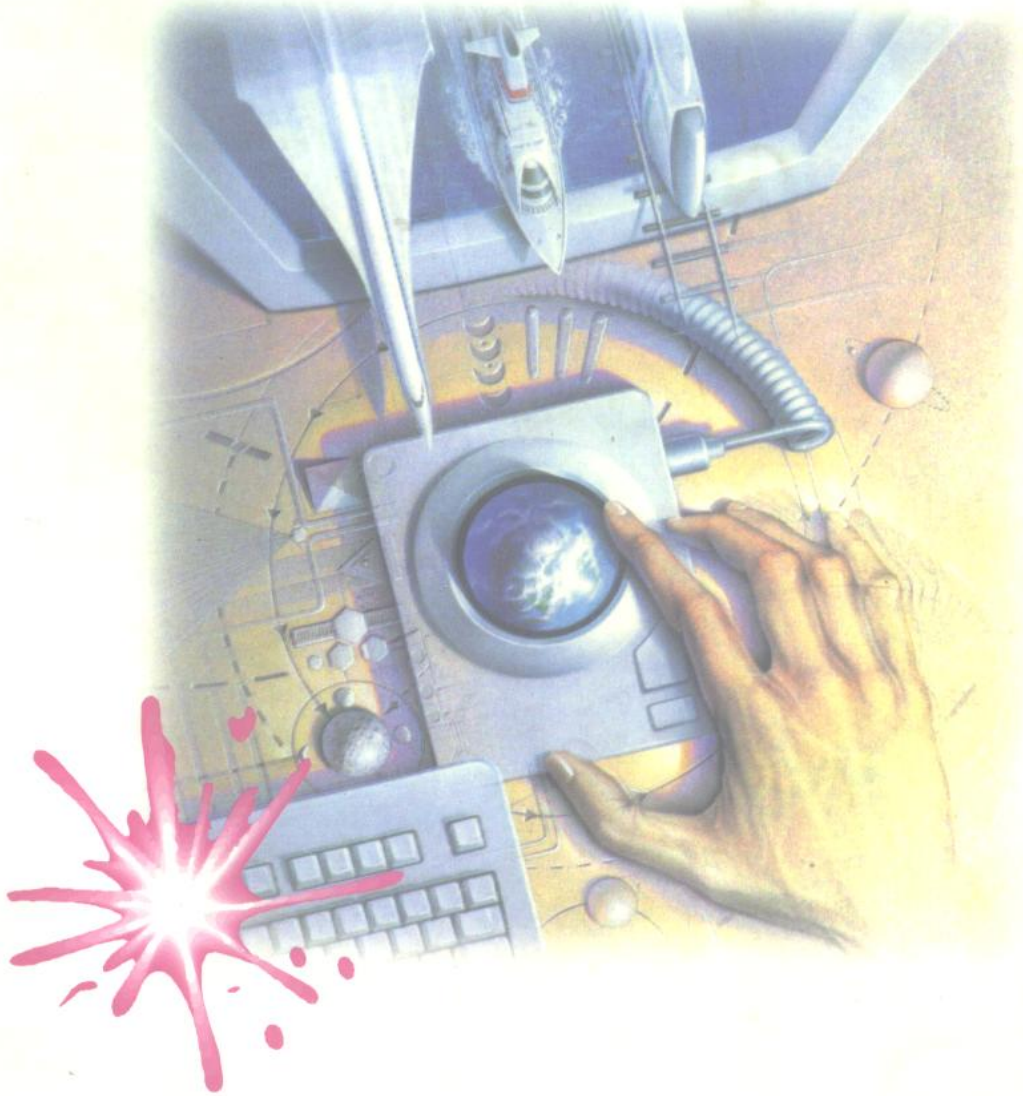
DELPHI 2.0

# 活学活用

# DELPHI 2.0

最新流行软件丛书



郭新明　曹晓阳 等 编

西南交通大学出版社

# 活学活用 Delphi 2.0

郭新明　曹晓阳
李济琛　欧　阳　编

西南交通大学出版社

## 内 容 提 要

本书介绍了如何使用 Delphi 2.0 进行程序设计。本书前二章介绍了使用 Delphi 2.0 进行 Windows 程序设计的基本知识和 Pascal 语言的基本内容。

在本书后面的章节中,介绍了使用 Delphi 2.0 进行 Windows 编程的常用技术(如 SDI 和 MDI 的程序设计技术)以及可视文件库(VCL)的设计;此外还介绍了如何在 Delphi 程序中使用 Windows QDI 函数和字体、如何在 Delphi 程序中进行打印。

本书还介绍了较为高级的多线性 Delphi 程序的编写、自定义文件的编写,以及多媒体播放器的使用和数据库程序的设计。

本书由浅入深地讲解了如何使用 Delphi 2.0 进行 Windows 95 环境下的程序设计,书中有大量可供读者实践的程序实例。阅读本书有助于掌握 Delphi 程序的开发设计。



**活 学 活 用 Delphi 2.0**
——最新流行软件系列丛书
郭新明 曹晓阳 等编
责任编辑 唐 晴
*
西南交通大学出版社出版发行
(成都 二环路北一段 610031)
郫县印刷厂印刷
*
开本:787 × 1092 1/16 印张:24.875
字数:626 千字 印数:1—5000 册
1997 年 7 月第 1 版 1997 年 7 月第 1 次印刷
**ISBN 7 - 81057 - 096 - X/T·267**
定价:48.00 元

# 目　录

# 第一章　使用 Delphi 2.0 进行 Windows 编程

Delphi 2.0 是 Windows 95 和 Windows NT 下的快速应用程序开发（RAD）和数据库开发的工具。Delphi 2.0 结合了可视化开发环境、快速有力并优化了的 32 位编译器和由紧密集成可调整的数据库引擎所提供的数据管理能力。Delphi 2.0 是唯一在一个开发环境无缝地集成了这 3 种主要技术的产品。

Delphi 2.0 针对不同的需要有 3 个不同的版本包装，分别是 Delphi 2.0 Desktop、Delphi 2.0 Developer 和 Delphi 2.0 Client/Server Suit。这 3 个版本分别针对不同的开发者。

Delphi 2.0 Desktop 是入门级版本，它提供了所有你开始使用 Delphi 2.0 编写应用程序所需要的一切。它包括以下特征：

- 优化了的 32 位 Object PASCAL 编译器
- 可视元件库（VCL），包括新的 Windows 95 控制和 100 个标准元件
- 面向对象的表单设计器，它支持可视的表单继承和链接
- Delphi 1.0，用于 16 位 Windows 开发
- 可与 Paradox 和 dBase 表连接的数据库元件
- 可用于 Paradox 和 dBase 表的 Database Explorer 工具
- 对 Win32 API 的全部支持，包括：OLE 自动化、OCX 控制、多线程和各种 Microsoft 和第三方软件开发工具(SDK)

Delphi 2.0 Developer 适用于非 Client/Server 的专业开发者。如果你是专业的应用程序或 Delphi 元件开发者，那这个版本就适合你。Developer 版本包括 Desktop 版本加上以下特征：

- 七个其它元件（共 107 个），包括数据绑定元件
- ReportSmith 报表工具，用于你的应用程序中集成化定制报表
- 单用户的本地 InterBase 服务器，使你能不必联入网络就可进行 Client/Server 开发
- 增强的数据库机能，包括：Database Explorer 工具、数据容器、对 ODBC 数据源的支持，低级 Borland 数据库引擎（BDE）API 存取
- InstallSHIELD Express 应用程序安装工具
- Open Tools API，紧密的集成于 Delphi 2.0 环境中的元件开发工具，和 PVCS 版本控制的接口
- 可视元件库（VCL）的源代码和 Object Pascal 运行库（RTL）
- WinSight32 工具，用于浏览窗口和消息信息

Delphi 2.0 Client/Server Suit 用于合作的 Client/Server 开发者。如果你是开发基于 SQL 数据库服务器的应用程序，在这个版本包含了适用于整个 Client/Server 开发周期的工具。Client/Server Suit 包括其它两个版本加上以下特征：

- 32 位本地化的 InterBase、Oracle、Microsoft SQL Server 和 Sybase 数据库服务器驱动程序
- SQL Database Explorer 使你能浏览和编辑特定服务器的数据
- SQL Monitor 使你能查看与服务器之间的 SQL 通讯，调谐你的 SQL 应用
- Data Pump Expert 用于快速升迁
- 两用户的 Windows NT 版 InterBase 许可证
- ReportSmith SQL 版本
- 集成的 PVCS 版本控制
- 可视的查询生成器
- 与 CASE 工具的集成

# §1.1 Delphi 2.0 集成开发环境

Delphi 2.0 集成开发环境分成四个主要部分：主窗口、表单设计器、对象观察器和代码编辑器。

## §1.1.1 主窗口

可以认为主窗口是 Delphi 2.0 集成开发环境的控制中心。主窗口拥有其他 Windows 程序主窗口的所有标准功能。它由 3 个部分组成：主菜单、快捷按钮条和元件板。如任何 Windows 程序，主菜单是你用于打开、创建或保存项目、表单、单元或文件。你也可以剪切、复制、粘贴或启动其他编程工具。快捷按钮条由可以访问主菜单常用功能的按钮组成。图 1.1 显示了 Delphi 2.0 的主窗口。


图 1.1 Delphi 2.0 的主窗口

元件板包含了代表 Delphi 2.0 元件的图标，如按钮、列表框、数据库和 OCX。快捷按钮条和元件板上的按钮都提供了“帮助提示”，当你把鼠标光标放在按钮上一或两秒就可以激活这些帮助提示。

**提示**：Delphi 2.0 的快捷按钮条是可配置的。在快捷按钮条上右击鼠标键，你就可以发现添加或删除按钮的下拉菜单选项。

## §1.1.2 表单设计器

表单设计器开始时是一个空窗口。它可以在你的艺术性的设计下变成 Windows 应用程序。使用表单设计器，你可以使用鼠标调整表单上元件的位置或大小，使用对象观察器和代码编辑器可以控制元件的外观和行为。

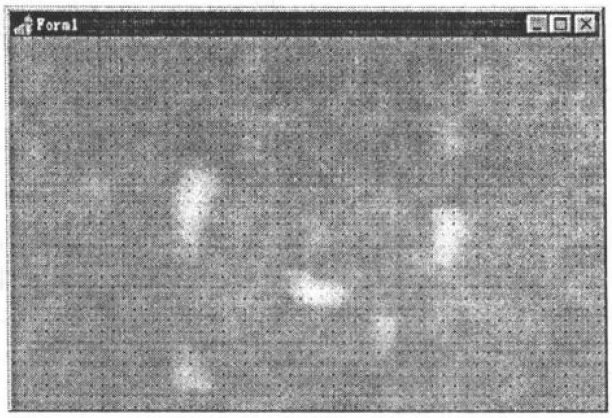



图 1.2　表单设计器

## §1.1.3　对象观察器

通过对象观察器，你可以修改表单或元件的属性或使你的表单或元件能对不同的事件做出反映。属性是指诸如高度、颜色和字体等决定对象外观的数据；事件是指你的应用程序对某个事件作出反映而执行的代码部分。鼠标按下消息和重画消息就是事件的两个例子。对象观察器窗口使用标准的 Windows 笔记本标签来进行元件属性或事件的切换，你只需要选择窗口顶端相应的标签即可，对象观察器中显示表单设计器中拥有聚焦的表单或元件的属性和事件。

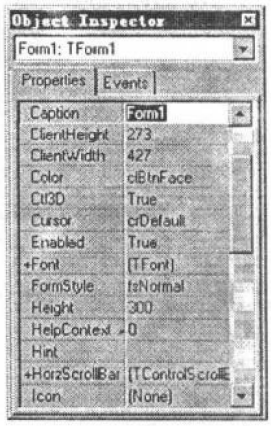



图 1.3　对象观察器

作为 Delphi 2.0 的程序设计员，你应当知道帮助系统是与对象观察器紧密相连的。如果你想了解某个特定的属性或事件，你只需要按下 F1 键，Winhelp 将显示所需的信息。

## §1.1.4　代码编辑器

代码编辑器是严格意义上真正编程的地方。在这里你可以键入你的代码，Delphi 2.0 也会在这里插入基于你应用程序中的元件所产生的代码。代码编辑器窗口顶端上包含笔记本标

签，它对应于不同的源代码模块或文件。每当在应用程序中添加一个新表单，就会创建一个新单元并在代码编辑器的顶端增加一个标签。你也可以在应用程序中添加无表单对应的单元模块，它们也会在代码编辑器中以标签形式出现。

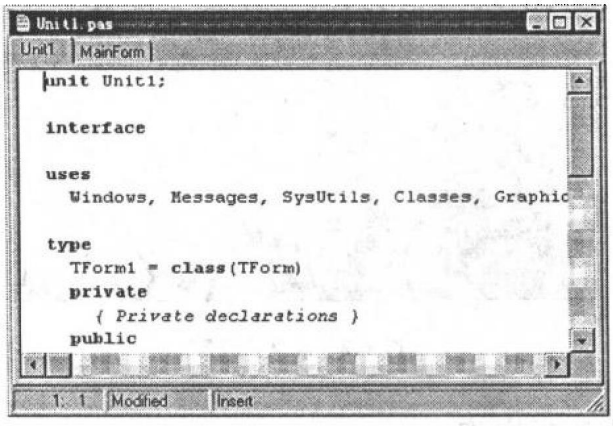



图 1.4　代码编辑器

## §1.2　源代码生成器

Delphi 2.0 集成环境会对你表单设计器中的可视元件生成 Object Pascal 源代码。例如，你在主窗口中选择菜单 File | New | Project 而开始一个新项目，在表单设计器中可以看到一个新表单，在代码编辑器中可以看到表单的源代码。源代码如下所示：

**清单 1.1　一个程序块**

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs;

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

var
  Form1: TForm1;

implementation

{$R *.DFM}

end.
```

值得注意的是与表单相关的源代码模块是存在单元中。每个表单都有一个单元，但并不

是每个单元都对应一个表单。在下一章中，我们将讲述 Pascal 语言，使你能对这段源代码有所了解。

其次，结构：

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

指示表单对象 TForm1 是由 TForm 派生出来的，在空白处你可以插入你自己的公共和私有变量。在下一章里，我们会详细解释什么叫对象、公共变量和私有变量。

{$R *.DFM}这一行也是很重要的，它指示 Pascal 加载外部资源文件。这一行将 DFM 文件链接到可执行文件中。DFM 文件包含了你在表单设计器中创建的表单的二进制代码。“*”是通配符，它代替与当前单元同名的文件。例如，上面例子中单元文件为 UNIT1.PAS 其中的 *.DFM 代表 UNIT1.DFM。

应用程序的项目文件，文件名是以 .DPR 结尾，实际上也是一个 Pascal 源代码文件。项目文件是你程序中的主要部分，下面是一个应用程序例子中的项目文件。

```
program Project1;

uses
  Forms,
  Unit1 in 'Unit1.pas' {Form1};

{$R *.RES}

begin
  Application.Initialize;
  Application.CreateForm (TForm1, Form1);
  Application.Run;
end.
```

如果在应用程序中添加更多的表单，它们将在项目文件中的 uses 子句中出现，相关的表单名将作为注释出现在 uses 子句的单元名字后面。如果你对单元对应哪个表单感到困惑，你可以选择菜单“View | Project Manager”打开项目管理器窗口来进行解答。

## §1.3 创建一个小的应用程序

一个简单的例子就是一个按钮元件添加到一个表单中，于是表单对象中添加如下代码：

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

现在，你可以看到按钮是 TForm1 类的一个实例变量。如果你在 TForm1 以外引用这个按钮，请记住指明 Form1．Button1，表示它是 TForm1 范围中的。

在表单设计器中选择按钮后，就可以通过对象观察器改变它的属性。例如：你想在设计时将按钮宽度改成 100 点，在运行时，你希望按下按钮后按钮会倍增其高度。要改变按钮的宽度，在对象观察器（Object Inspector）窗口中找到 Width 属性，然后将相应的值改成 100。要使按钮对鼠标单击事件做出反应，在对象观察器窗口中选择“Events”页，可以看到按钮能作出反映的事件列表。双击 OnClick 事件旁边的列，Delphi 2.0 将产生一个过程段，名称是 TForm1Button1Click。你可以在 begin 和 end 之间添加对事件做出反映的代码：倍增按钮高度：

```
Button1．Height ：＝ Button1.Height＊2;
```

按下键盘上的 F9 功能键，你就可以运行这个应用程序了。

在 Delphi 2.0 中事件通常是由 Windows 消息触发，例如 OnMouseDown 事件就是由 Windows 的 wm-XButtonDown 消息触发的。同时 OnMouseDown 事件方法中给出事件发生时哪个键被按下和鼠标所在的位置。Delphi 2.0 的事件可以代表几个不同的 Windows 消息，同时各个消息参数被转换成了易于理解的参数。你不必像标准的 Windows 消息处理一样，在事件处理上编写繁琐的代码。

# 第二章　Object Pascal 语言

本章将讲述 Delphi 2.0 所基于的语言——Object Pascal。在开始我们会介绍 Object Pascal 语言基本内容：语言规则和结构，然后你将学到 Object Pascal 中更高级的部分：类和异常处理等。读完这章后，你将明白一些编程的概念，如：变量、类型、操作符、循环、分支、异常和对象等。

## §2.1　Object Pascal 基本元素

### §2.1.1　注　释

开始，你将学到如何在你的 Pascal 代码中制作注释。Delphi 2.0 支持 3 种注释：花括号注释、花括号/星号注释和 C++类型的双斜杠注释，3 种注释的例子如下所示：

```
{用花括号括起来的注释}
{* 用花括号和星号括起来的注释 *}
//C++类型的注释
```

编译器将注释指示符之间的内容都当作是注释；对于 C++类型注释的行，从“//”后直到行结束都被认作是注释。

**注意**：建议不要使用嵌套注释，尽管嵌套不同类型的 Pascal 注释是符合语法的。

### §2.1.2　变　量

Object Pascal 要求你在过程、函数或程序开始之前声明所有的变量。例如：C++语言中的下面一段程序：

```
void foo (void)
{
        int x = 1;
        x++;
        int y = 2;
        float f;
        // ... etc ...
}
```

在 Delphi 中必须被结构化地写成：

```
procedure foo;
var
        x, y : integer;
```

```
        f : double;
begin
        x: =1;
        inc (x);
        y: =2;
        // ... etc ...
end;
```

**注意**：Object Pascal 不是大小写敏感的语言，大小写可任意使用。

Object Pascal 的语法结构使它的代码较 C ++ 或 Visual Basic 更易读和可维护。

Object Pascal 允许你将多个相同类型的变量放在一个行中定义，如下例中：

VarName1, VarName2 : SomeType ;

记住在 Object Pascal 中，声明的变量名是在类型之前，变量与类型之间有一个冒号。变量的初始化总是与变量的声明分开的。

Delphi 2.0 的一个新特征是允许你在 var 块中初始化全局变量。该语法的例子如下：

```
var
        I : integer = 10;
        P : Pointer = nil;
        S : String = 'Hello World';
        D : Double = 3.141579;
```

**注意**：只有全局变量才能预初始化，过程或函数的本地变量是不允许的。

## §2.1.3 常 量

Pascal 中的常量由 Const 子句定义，类似于 C 语言中的 Const 关键字。C 语言中的常量与 Object Pascal 中常量的定义主要区别是 Object Pascal 的常量不需要声明数据类型。Delphi 2.0 编译器根据常量的值自动分配合适的空间。例子如下：

```
const
        ADecimalNumber = 3.14;
        I = 10;
        ErrorString = 'Danger, Danger, Danger!';
```

你也可以指定所声明的常量类型，从而使你能控制编译器如何处理你的常量：

```
const
        ADecimalNumber : double = 3.14;
        I : Integer = 10;
        ErrorString : String = 'Danger, Danger, Danger!';
```

Object Pascal 允许在 Const 和 Var 声明中使用编译时间函数，如 Ord ()、Chr ()、Trunc ()、Round ()、High ()、Low () 和 SizeOf () 等。例如，下面的代码均是合法的：

```
type
        A = array [1..2] of integer;
const
        w : word = SizeOf (Byte);
var
        I : integer = 8;
```

```
j : SmallInt = ord ('a');
L : Longint = Trunc (3.14159);
x : ShortInt = Round (2.71828);
B1 : byte = High (A);
B2 : byte = Low (A);
C : Char = chr (46);
```

如果你试图改变常量的值，Delphi 2.0 将产生一个编译错误。因为常量是只读的，Object Pascal 可以在代码段存储这些常量从而优化你的数据空间。

**注意**：不像 C 或 C++，Object Pascal 没有预处理器，因此在 Object Pascal 中没有宏的概念，即没有与 C 语言中 #define 等价的常量声明。

## §2.1.4 操作符

操作符是你代码中用来对各种类型数据进行操作的符号。例如：对数值型数据有 +、-、* 和 /。

### 一、赋值操作符

赋值操作符“:=”用于给变量赋值，它等价于和 Visual Basic 中的“=”赋值操作符。例如：

```
Number1 := 5;
```

表示“给 Number1 赋予值 5”。

### 二、比较操作符

如果你用 Visual Basic 编过程序，那么对 Delphi 的比较操作符也应不感到陌生。因此本节将不作详细的介绍。

Object Pascal 使用“=”作为两个表达式或值之间的逻辑比较，等价于 C 语言中的“==”，于是 C 语言中

```
if (x == y)
```

在 Object Pascal 中将写成：

```
if x = y
```

Delphi 中的不等于操作符是<>，它等价于 C 语言中的!=操作符。

### 三、逻辑操作符

Pascal 使用保留字 or 和 and 作为逻辑操作符，C 语言使用||和 &&。

Pascal 的逻辑非操作符是 not，它等于 C 语言中的！操作符。逻辑操作符通常用于 if 语句或循环中。例如：

```
if (Condition1) and (Condition2) then DoSomeThing;
while not (Condition) do something;
```

### 四、数学操作符

你应当对 Object Pascal 的大部分数学操作符都已熟悉，因为它们与 C、C++ 和 Visual

Basic 中使用的数学操作符相同。它们分别是：

+：加

－：减

×：乘

/：浮点除

div：整除

mod：模

你可能注意到 Pascal 与其它语言不同，它对浮点数和整数有不同的除法。div 操作符自动截断除运算后的余数。

**注意**：对不同类型的表达式要使用正确的除法操作符。如果你试图用 div 操作符去除两个浮点数，或用/去除两个整数，Object Pascal 编译器将产生一个错误。如下例表示：

```
var
        I : Integer;
        r : real;
begin
        I : = 4/3;              //该行将引起编译错
        f : = 3.4 div 2.3       //该行也将引起编译错
```

### 五、位操作符

位操作符使你能够修改所给变量的某个位。通常位操作符有左移位和右移位、位与、位或、位非和位异或。它们分别是：Shl、Shr、And、Or、Not 和 Xor。

### 六、递增和递减操作符

递增和递减操作符会对将某个整数加 1 或减 1 产生优化的代码，Pascal 没有提供 C++ 中的++和－－操作符，但它的 inc（）和 dec（）操作过程将优化编译成一条机器指令。

调用 inc（）或 dec（）时你可以使用 1 个或 2 个参数。例如，下面两行代码：

```
inc (Variable);
dec (Variable);
```

要增减某个变量，方法之一就是使用 inc（）和 dec（）过程，例如：

```
inc (Variable, 3);
dec (Variable, 3);
```

它相当于用汇编指令 add 和 sub 对变量加 3 和减 3。

**注意**：当打开编译器优化时，inc（）和 dec（）与 Variable：＝variable＋1 产生相同的机器码，你可以按你的爱好选择使用。

## §2.1.5 Object Pascal 数据类型

Object Pascal 的最大特点是它们的强类型性，即传递到过程和函数的变量类型必须与过程或函数定义的参数类型相同。同样，Object Pascal 也不允许指针类型的不匹配（C 语言中将产生编译警告）。

## 一、数据类型对比

Delphi 的基本数据类型与 C 和 Visual Basic 的很相似，下表列出了 Object、Pascal、C 和 Visual Basic 基本类型的对比，该表可作为调用非 Delphi 2.0 的动态链接库（DLL）或目标文件（OBJ）中的函数时的类型匹配参考。

表 2.1 数据类型对比

| 变量类型 | Pascal | C | Visual Basic |
|---|---|---|---|
| 8 字节有符号整数 | Short | Int | short |
| 8 字节无符号整数 | Byte | BYTE, unsigned short | |
| 16 字节有符号整数 | SmallInt | int | Integer |
| 16 字节无符号整数 | Word | unsigned int | |
| 31 字节无符号整数 | Cardinal | | |
| 32 字节有符号整数 | Integer, Longint | long | Long |
| 32 字节无符号整数 | unsigned | long | |
| 64 字节有符号整数 | int64 (VC++) | | |
| 4 字节浮点数 | Single | float | Single |
| 6 字节浮点数 | Real | | |
| 8 字节浮点数 | Double | double | Double |
| 10 字节浮点数 | Extended | long | double |
| 64 字节货币 | Currency | Currency | |
| 16 字节可变数据 | Variant, TVarData | VARIANT | (Default) |
| 1 字节字符 | Char | char | |
| 2 字节字符 | WideChar | | |
| 固定长度字符串 | ShortString | | |
| 动态长度字符串 | AnsiString | $ | |
| Null 结尾的字符串 | PChar | char far * | |
| Null 结尾的广义字符串 | PWideChar | LPCWSTR | |
| 1 字节布尔类型数据 | Boolean, ByteBool(任意 1 字节) | | |
| 2 字节布尔类型数据 | WordBool (任意 2 字节) | Boolean | |
| 4 字节布尔类型数据 | BOOL, LongBool | BOOL | |

**注意**：Delphi 2.0 的整数类型是 32 位的。范围由 (-214783648 到 214748364)。

## 二、字符类型

Delphi 2.0 提供了 3 种字符类型：

AnsiChar　标准的 1 字节 ANSI 字符

WideChar　Delphi 2.0 的类型，双字节代表 Unicode 字符