开发专家之Delphi

飞思科技
FECIT Sci-Tech
www.fecit.com.cn

Delphi 6

开发者手册

飞思科技产品研发中心 编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

开发专家之 Delphi

Delphi 6 开发者手册

飞思科技产品研发中心　编著

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书属于《开发专家之 Delphi》系列。本书是为程序员准备的速查手册，专业的程序员凭借本书的信息支持就能开发出最标准和完美的 Delphi 程序。

全书共分 6 章，包括语言参考、系统常量、常见错误信息和 Delphi 代码开发标准。列出了 Delphi 开发中的常用函数、过程、变量、类、方法和属性，供编程者查阅。

本书适合于作为中高级 Delphi 程序员的编程参考书。

图书在版编目（CIP）数据

Delphi 6 开发者手册/飞思科技产品研发中心编著. —北京：电子工业出版社，2002.1
（开发专家之 Delphi）
ISBN 7-5053-7358-7

Ⅰ.D... Ⅱ.飞... Ⅲ.Delphi 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2001）第 093140 号

丛 书 名：开发专家之 Delphi
书　　名：Delphi 6 开发者手册
编　　著：飞思科技产品研发中心
责任编辑：郭　晶　罗建强
排版制作：电子工业出版社计算机排版室监制
印 刷 者：北京大中印刷厂
出版发行：电子工业出版社　http://www.phei.com.cn
　　北京海淀区万寿路 173 信箱　邮编：100036
经　　销：各地新华书店
开　　本：787×1092　1/16　印张：26.25　字数：672 千字
版　　次：2002 年 1 月第 1 版　2002 年 1 月第 1 次印刷
书　　号：ISBN 7-5053-7358-7 / TP・4239
定　　价：35.00 元

出 版 说 明

“开发专家”是电子工业出版社计算机图书研发部长期以来精心培育的计算机科学技术类本版品牌。这个品牌是由多个专题系列组成的横向大系列，涵盖了计算机技术的各个方面，特别是一直受到极大关注的程序开发类系列，例如《开发专家之数据库》、《开发专家之网络编程》、《开发专家之 Delphi》以及《开发专家之 Sun ONE》等。这些专题系列基于各自的角度，从纵向上包含了该专题的所有内容。因此，整个“开发专家”的品牌架构纵横交错，囊括了所有的计算机技术和所有的技术层面，海纳百川而又极具可扩展性。

“开发专家”的作者队伍主要依托于“飞思科技产品研发中心”。“飞思科技产品研发中心”由专业的策划人员、权威的技术专家和资深的作者队伍共同构成。在图书的出版上，形成了以研发为基础、以出版为中心、以服务为支持的专业化出版框架和流程。通过深入的市场调查和技术跟踪，在综合了技术需求和读者焦点等因素的基础上，形成各系列丛书的写作重点和大纲，然后聘请业界的最前沿学者进行写作。同时，策划工作全程介入写作进程，严格控制写作质量，用最专业的技术背景、最深刻的理论基础、最具代表性的案例、最能为专业读者接受的形式，为读者提供品质最佳的图书产品，体现了出版者和著作者的完美结合。

多年来，计算机图书研发部始终把创造社会效益摆在首位，秉承一切为国内计算机技术专业读者服务的精神，为推动国内 IT 技术发展、为体现国内技术的原创水平，穷尽所有的创意与努力，将出版者的命运与读者的支持紧紧地连在了一起。

在此，我们临出版之残酷竞争而不惧，旌旗猎猎而异军突起，这与广大读者的支持是分不开的。为使我们的脚步更坚实、使我们的队伍永远保持活力和创造力，我们期待着您能为我们的前进贡献出您的意见和建议。同时，我们也在等待着您的加入。

我们的联系方式：

电　话：　（010）68134545
E-mail：　support@fecit.com.cn
网　址：　http://www.fecit.com.cn　http://www.fecit.net

电子工业出版社计算机图书研发部

前　言

关于本套丛书

Delphi 是美国 Borland 公司推出的功能强大的应用程序开发工具。它具有功能强大、运行速度快、易于学习和使用以及开发效率高等特点。它是可视化应用编程开发环境、可重用性面向对象编程语言、快速编译器和数据库的完美结合。

Delphi 6 发布于 2001 年 6 月。新版本的 Delphi 6 开发功能更加强大，除了能够有效帮助开发者个人或其开发团队快速建立 Windows 应用程序，快速简化 Windows 与浏览器客户、Web 服务器、中间件以及后台数据库系统的集成等等这些传统意义上的开发之外，Delphi 6 是目前惟一全面支持所有主要工业标准（XML、SOAP、WSDL、XSL 等）的开发工具，同时也支持基于 Web 服务的 Microsoft .NET 和 Sun ONE 体系，而且提供给 Web 开发者需要的可伸缩性与可靠性。新的 Delphi 6 框架中还包括了 BizSnap、WebSnap 和 DataSnap，用户可以用它开发支持 Web 服务特性的服务器端和客户端应用，而这一切都是通过一套高度集成的可视化开发工具、先进的编译技术和可重用的组件完成的。特别是在电子商务愈加流行的今天，通过 Delphi 6，任何企业都能很快地转移到未来基于 Web 服务的电子商务应用程序开发上，而不用丢弃以往的开发方式、技巧和源代码。可见，要创建一流的 Web 应用程序，Delphi 6 无疑是目前的最佳选择之一。

《开发专家之 Delphi》系列丛书就是针对 Delphi 6 的整个开发体系和特色进行组织的，涵盖了 Delphi 开发所有重要方面，为开发人员提供了完整的知识架构，无论要进行怎样的 Delphi 开发，都可以在这套丛书中找到明确的解决方案和经验之谈。

同时，专业的作者队伍、完整的解决方案和详尽的实例剖析是这套丛书高质量的基础和保证，也是本套丛书的最大特色所在。

总之，哪怕是最熟练的程序员也需要专业的技术文献，这套丛书就是经典的开发经验及实例的集成，所以必将受到专业人士的关注和欢迎。

关于本书

《Delphi 6 开发者手册》是为专业程序员准备的开发速查手册。

本书可以帮助读者方便快捷地掌握 Delphi 6 的新特性和编程方法，大大缩短开发应用软件的时间，提高开发效率。

全书共分 6 章。

第 1 章“Object Pascal 速查”，介绍了 Delphi 6 中的一些基本概念：常量、变量、数据类型、运算符、函数和过程、单元、基本语句、文件 I/O 和包，是本书的基础。

第 2 章“Delphi 对象速查”，介绍了对象、类、消息、接口、内存管理、线程、TThread 对象等内容。

第 3 章“Delphi 语言参考速查”，这是本书的核心部分，它按字母顺序列出了 Delphi 6 的关键字、指示字、函数、过程、变量、类、方法和属性，并提供完整的例子展示如何正确有效地使用 Delphi 语言。

第 4 章“编译器指示字速查”，介绍了开关型编译器指示字、参数型编译器指示字和条件型编译器指示字。

第 5 章列出了“常见的错误信息”，包括编译错误信息和运行错误信息，其中运行错误信息又分为 3 类：I/O 错误、致命错误、操作系统错误。

第 6 章“Delphi 代码开发标准”介绍了 Delphi 代码的开发标准。

本书由飞思科技产品研发中心策划并组织编写，李辉等参加了本书的写作。同时，在本书的写作过程中得到了刘文智先生的大力支持和协助，他提出了大量的参考性意见更使本书增色不少，在此表示衷心的感谢。

当然，限于作者水平，加之时间仓促，书中不足之处难免，敬请读者批评指正。

我们的联系方式：

电　话：　（010）68134545　68134811

E-mail：　support@fecit.com.cn

网　址：　http://www.fecit.com.cn　http://www.fecit.net

飞思科技产品研发中心

目　录

第 1 章　Object Pascal 速查

Delphi 的编程语言是以 Pascal 为基础的。Pascal 语言具有可读性好、编写容易的特点，这使得它很适合作为基础的开发语言。同时，使用编译器创建的应用程序只生成单个可执行文件（.EXE），正是这种结合，使得 Pascal 成为 Delphi 这种先进开发环境的编程语言。

1.1　常　量

顾名思义，常量的值在程序执行过程中是不可改变的。引入常量的目的是为了给某些难记的数字一个容易记忆的符号，同时，使用常量也可以更加容易地修改程序。

常量通过关键字 Const 来声明，Object Pascal 语言不需要专门指定常量的数据类型，编译器会根据常量的值自动判断常量的类型并分配内存。常量可以是数字、字符或字符串，示例如下：

```
const
    pi = 3.1415926
    number = 1000000
    book = 'Delphi'
```

常量也有类型，与变量不同的是，常量假设其类型就是常量说明中其所代表的值的类型。上文的三个常量的类型分别是 real 型、整型、字符串型。常量用“=”表示两边的值是相等的。

1.2　变　　量

变量在程序代码中代表一个内存地址标识符，而此地址的内存内容可以在程序代码执行时被改变。每个变量都有一个名字和数据类型，名字可用来引用变量，而数据类型则决定该变量的取值范围。

1.2.1　变量的声明

在使用变量前必须对它进行说明，即对它进行命名，并说明它的类型。在所有变量说明以前应加上保留字 var。变量声明左边是变量的名称，右边则是该变量的数据类型，中间用“：”隔开。示例如下：

```
var
I , g : Interger;
```

```
Message : String;
```

在窗体中加入一个名称为 Edit1 的编辑框，再加入一个名称（属性 Name）为 Add 的按钮部件，并建立如下的事件处理过程：

```
procedure TForm1.addClick(Sender: TObject);
var
X , Y: Integer;
begin
X := 100;
Y := 20;
Edit1.Text := IntToStr(X + Y);
end;
```

在本例中，当单击 Add 按钮时，编辑框中显示值为 120。在 Object Pascal 中，必须确保变量或属性被赋予类型相同或兼容的值。在本例中可以尝试将赋给 X 的值改为 1 000，或去掉 IntToStr 函数，在编译时将会出现类型不匹配的错误，这也说明了 Object Pascal 强制类型转换语言的特点。

声明变量要注意两点，一是变量名要符合 Object Pascal 语言关于标识符的规则；二是必须明确指定变量的类型，类型要么是 Object pascal 语言预定义的标准类型，要么是前面已声明过的自定义类型。

程序示例如下：

```
Type Digits=Set of 0..9;
Var
X,Y,Z:integer;  {X，Y，Z 类型相同，把它们合并在一起声明}
M,N:Digits;     {Digits 类型已经在前面声明}
```

1.2.2 Absolute 子句

声明变量时还可以带一个可选的 Absolute 子句，用于指定变量的值在内存中存贮的绝对位置。Absolute 子句的语法如下：

```
Absolute 无符号整型：无符号整型
```

这个指示字后可以跟一个段值和一个偏移量，如：

```
Var CrtMode:Byte Absolute $0040:$0049;
```

在上例中，声明了一个 BYTE 类型的变量 CrtMode，这个指示字后第一个常数是段值，第二个常数是偏移量，段和偏移量只能在 $0000 至 $FFFF 之间。

这种形式不能用于 Windows 的保护模式下，因为 Windows 不允许应用程序访问程序外的内存区域。

直接写入内存地址会导致一个 GPF 错误。实际上，在 Windows 中，一般是声明另一个变量，然后在这个子句中指定变量的值的存贮地址跟该变量相同，例如：

```
Var Str:String;
```

```
StrLen:Byte Absolute Str;
```

在上例中，先声明了 String 类型的变量 Str，然后声明了一个 Byte 类型的变量 Strlen，Absolute 子句指定变量 StrLen 的地址与变量 Str 的地址相同。

1.2.3　全局变量和局部变量

全局变量分为两类，一是整个程序都能访问的公共变量，必须在单元的 interface 部分声明；另一类是只限于某个单元访问的公共变量，必须在该单元的 Implementation 部分声明。

对于全局变量，可以在声明的同时赋一个初始值，例如：

```
Var X:integer=1;
```

没有显式地赋初值，这个全局变量的初始值就是 0。

局部变量的作用域只限于声明所在的块内，通常是在过程、函数或类的方法内部声明，例如：

```
Procedure MyProc;
Var X,Y,Z:integer;
Begin
        ...
End;
```

上例中，X，Y，Z 只是在过程 MyProc 内有定义。

对于局部变量而言，不能在声明时赋初值，在明确地给它们赋值之前，它们的值是不确定的。

1.3　数 据 类 型

Object Pascal 语言的最大特点是，它对数据类型的要求非常严谨，也就是说，传递给过程或函数的参数值必须与形参的类型一致。在 Object Pascal 语言中，不会像 C 语言程序员那样经常碰到遇到诸如“可疑的指针转换”等警告信息。正是因为 Pascal 的数据类型比较严谨，它会对代码进行严格检查，以保证不会有错误。

1.3.1　简单类型

Object Pascal 中的简单类型包括整型、字符型、布尔型、枚举型、子界类型以及实数类型。其中整型、字符型、枚举型、子界类型统称为有序类型。除了整型，每种类型都有惟一的前驱；除了子界类型外，每种类型都有惟一的后继。对整型而言，其顺序就是该值本身；而对子界类型以外的其他类型，第一个值的顺序为 0，第二个为 1，依次类推。

整型

Object Pascal 中提供的各种整型说明如表 1-1 所示。

表 1-1 整数类型

类 型 名	范 围	备 注
Integer	-2 147 483 648～2 147 483 647	32 位带符号数
Cardinal	0～42 994 967 295	32 位无符号数
Shortint	-128～127	8 位带符号数
Smallint	-32 768～32 767	16 位带符号数
Longint	-2 147 443 648～2 147 483 647	32 位带符号数
Int64	-2^{63}～2^{63}-1	64 位带符号数
Byte	0～255	8 位无符号数
Word	0～65 535	16 位无符号数
Longword	0～4 294 967 295	32 位无符号数

对整型数进行算术运算时要特别注意数据的字长和值域。Object pascal 规定：对于整数常量，其精确类型是包含该值在内的值最小的类型。两个不同类型的操作数进行算术运算时，首先把它们转换成包含两种类型所有值在内的值域最小的类型。

在赋值语句中，赋值号右边的表达式的计算独立于赋值号左边的变量的类型。

可以用类型强制转换把一种整数类型转换成另一种整数类型。

对于整数常数，还可以用十六进制表示，表示方法是在数的前面加 $ 符号，例如$10 在十进制中就是 16，$3C 在十进制中就是 60。

字符型

基本的字符类型有 AnsiChar 和 WideChar 两种，其中 AnsiChar 的值占用 8 位，根据扩展 ANSI 字符表排列；而 WideChar 的值占用 16 位，根据 Unicode 码表排列，前 256 个 Unicode 字符同 ANSI 字符相同。

在 Delphi 中通用的字符类型为 Char。对于任何在 AnsiChar 或 WideChar 范围内的整数值，可以用预定义的函数 Chr 返回对应的字符值。和整数一样，在没有启用界限检查的情况下，字符值是环绕的——越过上界后会自动返回下界。

如果程序中必须用到字符的字节数，最好先用 SizeOf 函数取得。

Object Pascal 提供了两个函数用于对字符类型操作：Ord 用于返回一个字符的序号，Chr 用于把一个整数转换成相应序号的字符。

这里还要介绍控制字符的概念。Object Pascal 中可以用一个 # 后紧跟一个 0~255 的整数常数作为相应 ASCII 码的字符，例如，#0 表示 ASCII 码中序号为 0 的字符即空字符；#27 表示 ASCII 码中序号为 27 的字符即 ESC。可以把控制字符作为普通字符和字符串结合在一起，例如："My Name Is'#49#49#49'Xu"。字符串中加了 3 个#49 字符表示 3 个 1。

整数常数必须紧跟在 # 之后，中间不能有空格。另外，如果几个控制字符是字符串的一部分，控制字符之间也不能有空格。

布尔型

布尔类型包括 Boolean，ByteBool，WordBool 和 LongBool 四种数据类型。Boolean，ByteBool 各占用一个字节，WordBool 占用两个字节，LongBool 占用四个字节。

对于 Boolean 来说，其值只能是 0（False）或 1（True），而 ByteBool、WordBool 和 LongBool 可以是有符号的整数，值为 0 代表 FALSE，非 0 值代表 TRUE。

其中最常用的是 Boolean 类型，而 ByteBool、WordBool 和 LongBool 类型则是为了与其他语言和 Windows 环境兼容，因为 Windows 的 API 在返回一个布尔值时，其值可能是一个两字节的有符号整数。如果试图把返回值赋给 Boolean 类型的数据，编译器认为类型不匹配，如果进行类型强制转换，又可能使返回值的有效数据被截断。

另外，Object Pascal 规定：False 小于 True，Ord（False）等于 0，Ord（True）等于 1，Succ（False）等于 True，Pred（True）等于 False。

枚举类型

枚举类型和子界类型不同于标准的简单类型。标准的简单类型如整型和实型等，其字长和值域以及所能参加的运算都已由 Object Pascal 语言预先定义好，程序员不必再进行类型描述。而枚举类型和子界类型则是由用户自定义的，同样是枚举类型，如果类型描述不同，这个类型的数据所表示出来的特性也不同。

所谓枚举类型，就是用一组数量有限的标识符来表示一组连续的整数常数，使用枚举类型能够更清晰地表示出现实世界。例如：一个星期有 7 天，程序中分别用 0、1、2、3、4、5、6 来表示一个星期的每一天，当然也可以，但程序很不直观，也许您在想，用常量来表示不也可以吗？是的，不过用枚举类型有其独到之处。

声明一个枚举类型的语法是这样的：

变量名＝（枚举列表）

枚举列表：标识符 1，标识符 2…标识符 n

其中，标识符列表中的标识符彼此之间用逗号隔开，它列出了枚举类型所能取值的范围。也就是说，一个枚举类型的变量，其值总是所列的标识符列表中的一个，这也是这种类型之所以叫枚举类型的原因，这是枚举类型的一个特点。例如：

```
Type
    DayOfWeek=(Sun,Mon,Tue,Wed,Thu,Fri,Sat);
```

Object Pascal 规定，第一个标识符的值为 0，第二个标识符的值为 1，依次类推。

上例中只是声明一个枚举类型 DayOfWeek，但类型不能直接在程序中参加运算。好像 Integer 类型一样，还必须声明一个 DayOfWeek 类型的变量，例如：

```
Var  MyDays:DayOfWeek;
```

这样就声明了一个 DayOfWeek 类型的枚举变量 MyDays，MyDays 的值总是标识符列表中的一个。

当然，为了简化程序，可以把类型声明和变量声明合二为一，例如：

```
Var MyDays:(Sun,Mon,Tue,Wed,Thu,Fri,Sat);
```

声明枚举类型和枚举变量时要注意：

枚举的元素只能是标识符，标识符的命名必须符合 Object Pascal 关于标识符的规定。

例如下面的声明就是错误的：

```
Type DayOfWeek=(0,1,2,3,4,5,6,7);（标识符不能以数字打头）
Type DayOfWeek=(For,Of,End,Do)（标识符不能是保留字）
```

同一个枚举元素不能同时出现在多个枚举中，例如：

```
Type Colors=(Red,Blue,Green,White,Black);
AnotherColors=(Yellow,Line,Silver,Green);
```

上面的两个类型中都有 Green 元素，这是不允许的。

不能直接用枚举元素参加运算，以枚举类型变量 MyDays 为例，程序中如果出现 X:=Fri*4 则是错误的，只能用某个枚举元素对枚举变量赋值，例如：

```
MyDays:=Fri;
```

Object Pascal 中提供了几个标准的函数用于对枚举变量进行操作，其中 Pred 函数用于返回指定元素的前一个元素，Succ 函数用于返回指定元素的后一个元素，例如（以 MyDays 变量为例）：Pred（Fri）等于 Thu，Succ（Sun）等于 Mon。

此外，还有一个函数 Ord 用于返回指定的枚举元素的值（序号），例如：Ord（Wed）等于 3。

子界类型

枚举类型的特点在于它的值是屈指可数的，而子界类型的特点则在于子界类型是有序类型中某范围内的值，主要用于限制一个变量的取值范围（把值全部罗列出来可能有困难）。

声明一个子界类型的语法是这样的：

```
子界类型＝下界常数..上界常数
```

其中，下界常数和上界常数的类型必须是同一种有序类型，如整型、布尔型、字符型、还可以是枚举型，但不能是实型，这两个常数的类型也称为子界的宿主类型。

程序示例如下：

```
Type Letters='A'..'Z';
     Month=1..12;
```

子界的宿主类型也可以是枚举型，不过在声明宿主类型为枚举型的子界类型之前必须先声明一个枚举类型，例如：

```
Type Month=(Jan,Fed,Mar,Apr,May,Jun,Jul,Ang,Sep,Oct,Nov,Dec);{声明一个枚举类型}
     Months=Jan..Dec;{声明一个子界类型}
```

跟枚举类型一样，类型是不能直接参加运算的，还必须声明一个子界类型的变量，例如：

```
Var MyMonths:Months;
```

为了简化程序，也可以把类型声明和变量声明合二为一，如：

```
Var MyLetters:'A'..'Z';
```

声明和使用子界类型要注意：上界常数和下界常数的类型必须一致，且都是有序类型。子界型变量所有参加的运算可参照它的宿主类型。子界型变量的取值是有范围的，如果运算过程中超出这个范围将导致程序出错。

Delphi 的文档中提到：子界型的上界和下界也可以是表达式。

实型

实型不是有序类型，因为它的值域是无限的，实型类型的数据也叫实数，实型用于表示通常所说的浮点数。实型的表示范围与存储格式如表 1-2 所示：

表 1-2　实数类型

类 型 名	范　围	有 效 位	占用内存（子节）
Real48	2.9×10^{-39}～1.7×10^{38}	11～12	6
Single	1.5×10^{-45}～3.4×10^{38}	7～8	4
Double	5.0×10^{-324}～1.7×10^{308}	15～16	8
Extended	3.6×10^{-4951}～1.1×10^{4932}	19～20	10
Comp	$-2^{63}+1$～$2^{63}-1$	19～20	8
Currency	-922337203685477.5808～922337203685477.5807	19～20	8
Real	5.0×10^{-324}～1.7×10^{308}	15～16	8

其中，一般较常用的是 Real，Single，Double。

实数也可以用科学表示法表示，例如：

7E-2 表示是 7 乘以 10 的负 2 次方，其中 7 称为尾数部分，-2 称为指数部分。

12.25E+6 或 12.25E6 表示是 12.25 乘以 10 的 6 次方。

使用科学表示法要注意：指数必须为整数，可以为正也可以为负，即使尾数为 1，也不能省略，如写成 E-2 就是错误的。

1.3.2　字符串类型

Object Pascal 的字符串类型分为短字符串和长字符串两种。

所谓短字符串，是指字符串长度最大不超过 255 个字符的字符序列。当编译开关的状态为{$H-}时，用保留字 String 声明的是一个短字符串，不管编译开关$H 在什么状态，用 ShortString 声明的变量总是短字符串类型，程序示例如下：

```
Var MyString1:ShortString;
Var MyString2:String[100];
```

上例中，保留字 String 后用方括号括起来的无符号整数，表示字符串的最大长度是 100。当把一个字符串赋给一个短字符串类型的变量时，超过其最大长度的部分将被截掉。

短字符串中的每一个字符可以通过字符串名加字符索引来访问。注意，尽管索引是从 0 开始的，但索引 0 处的字符存放的是字符串的实际长度，也就是说调用 Ord(S[0])相当于调用 Length(S)。真正的字符是从索引 1 处开始的。

由于短字符串的长度是动态变化的，可以使用 Low 和 High 函数取得字符串的最大序号和最小序号，Low 的返回值当然是 0，而 high 的返回值就是声明的字符串的最大长度。

短字符串类型主要是为了与早期 Delphi 版本兼容，在 Delphi1.0 中，并没有长字符串