Delphi程序库

Delphi 是一个面向对象的可视化程序开发环境，系统提供的编程语言 Object Pascal 既有结构化程序语言的各种功能和特征，又具备了面向对象的程序语言（OOPL）的继承、封装、多态以及软件重用等特征。Delphi Object Pascal 不但提供了丰富的运行时间库 RTL，还提供了大量的可重用的软件模块（component，组件），本章将对 Object Pascal 中的运行时间库 RTL 和可视组件库 VCL 进行介绍。

本章介绍的主要内容包括：

（1）Delphi 程序库组成

（2）构成 Delphi 程序库的程序单元及各单元中定义的内容

（3）运行时间库 RTL 内容列表

（4）可视组件库 VCL 分类列表

（5）可视组件库 VCL 层次结构详解

1.1 Delphi 程序库构成

Delphi 程序库包含两部分内容，即运行时间库 RTL 和可视组件库 VCL，Delphi 程序人员使用程序库的这两个部分构造应用程序。

1.1.1 运行时间库RTL

运行时间库 RTL 是所有 Object Pascal 程序都可以调用的函数

和过程的集合，是 Delphi Object Pascal 程序库的基本组成部分。

运行时间库 RTL 由一系列常用的基础函数、过程、常量、变量定义组成，运行时间库同时还包含处理程序运行异常（exception）的各种异常类。Object Pascal 运行时间库 RTL 给应用程序和可视组件库 VCL 提供了最基本的支持。

在 Object Pascal 中，运行时间库中函数、过程、常量和变量的定义都位于 System 单元中，运行时间库中的各种异常类定义位于 SysUtil 单元中。

1.1.2 可视组件库VCL

Delphi 可视组件库 VCL（Visual Component Library）是 Delphi Object Pascal 提供的面向对象的可重用软件模块，是一个功能强大的程序对象库。在 VCL 中，包含基本的程序对象，其中大多数对象是可视化的组件，使用这些对象和组件，可以极大提高编写 Windows 程序的效率，Delphi 本身，包括集成开发环境 IDE，常用工具等，都是使用 VCL 编写实现的。

Delphi Object Pascal 提供的可视组件库 VCL 是一个完全的面向对象的程序库，即 VCL 库是一个对象库，类似于 Borland C++中的 OWL 或 Microsoft Visual C++的 MFC。

VCL 中的大部分对象称为组件（component），从总体上划分，VCL 对象可分为基础对象和组件两大部分；在 VCL 的组件中，许多组件在程序运行时是可见的，这些组件也称为控件或控制（control）。

大多数控件在运行时可以接收程序的焦点（focus），即可以接收 Windows 消息，这类控件称为窗口控件（windowed control，基于窗口的控件），与窗口控件对应，那些不能接收焦点的控件称为非窗口控件（nonwindowed control），这类控件不能接收 Windows 消息，也不能包含其它的控件。图 1-1 简单描述了 Delphi VCL 库的组成和划分。

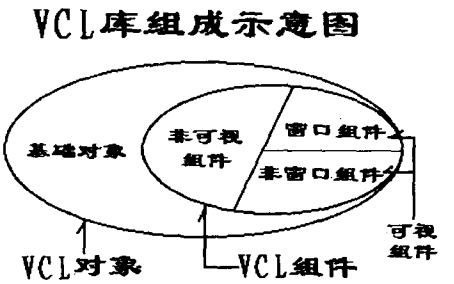

图1-1 Delphi VCL库组成

Delphi 程序库是 Delphi 应用程序开发的基础，下面，我们给出一个简单的示意图描述应用程序与 RTL、VCL 以及 Object Pascal 语言的关系，如图 1-2 所示。

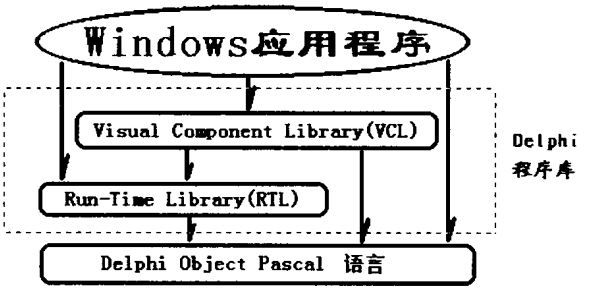

图1-2　应用程序与Delphi程序库的关系

前面我们简单介绍了 Delphi Object Pascal 程序库的构成，在后面的章节中，我们将详细介绍 Delphi Object Pascal 程序库的各项内容。

1.2 RTL函数、过程常量和变量

Delphi 运行时间库 RTL 是 Delphi 程序库的基础组成部分。RTL 包含的函数、过程、常量等都是在 System 程序单元中定义的，System 单元自动被所有其它程序单元引用，所以，在任何 Object Pascal 程序单元中，我们不必在程序的 USES 语句中引用该单元，就可以自由地调用 RTL 库中的函数和过程，使用 RTL 库中的变量和常量定义。

在第二章中，我们将详细介绍各常量、变量、函数和过程的具体含义和用法，本节中，我们只给出 RTL 的内容列表。

1.2.1 RTL常量和变量

Delphi 运行时间库 RTL 中定义的常量和变量如表 1-1 和表 1-2 所示。

表 1-1　　Delphi 运行时间库中的类型常量定义

常量名称	类 型	初 值	说 明
CmdLine	PChar	nil	命令行信息指针
CmdShow	Integer	0	CreateWindow 函数的 CmdShow 参数值
ErrorAddr	Pointer	nil	运行时间错误例程地址
ErrorProc	Pointer	nil	错误处理过程
ExceptDebugPtr	Pointer	nil	TuboDebugger 异常挂接指针
ExceptionClass	TClass	nil	异常基类
ExceptList	Word	0	异常处理句柄列表
ExceptProc	Pointer	nil	未捕获异常的异常处理句柄
ExitCode	Integer	0	程序退出码
ExitProc	Pointer	nil	退出处理例程
FileMode	Byte	2	文件打开模式常量
AllocFlags	Word	2	堆空间分配标志（gmem_Moveable）
HeapBlock	Word	8192	堆块尺寸
HeapCheck	Pointer	nil	堆空间一致性检查挂接地址

续表

常量名称	类 型	初 值	说 明
HeapError	Word	nil	堆错误函数
HeapLimit	Word	1024	堆块空间最小尺寸值
HeapList	Word	0	堆段列表
HInstance	Word	0	本程序当前执行副本（instance）标识
HPrevInst	Word	0	程序前一执行副本标识
InOutRes	Integer	0	I/O 结果状态
PrefixSeg	Word	0	程序段前缀(PSP)
RaiseList	Word	0	异常出发列表
RandSeed	LongInt	0	随机数发生器初值（种子）
SelectorInc	Word	0	地址选择器（Selector）增量
Test8086	Byte	0	处理器测试结果值
VtInteger	0		数据类型代码
VtBoolean	1		数据类型代码
VtChar	2		数据类型代码
VtExtended	3		数据类型代码
VtString	4		数据类型代码
VtPointer	5		数据类型代码
VtPChar	6		数据类型代码
VtObject	7		数据类型代码
VtClass	8		数据类型代码

表 1-2 Delphi 运行时间库中的变量

变量名称	类 型	说 明
Input	Text Input	标准输入文件
Output	Text Output	标准输出文件

1.2.2 函数和过程

在本节中，我们分类给出 RTL 库的所有函数和过程。RTL 中的函数和过程分为 18 个类别，分别列于表 1-3 到表 1-20 中，如下所示。

1.2.2.1 数学类例程

RTL 提供的数学类例程如表 1-3 所示。

表 1-3 数学类例程

名 称	功能说明
Abs	取绝对值
ArcTan	反正切三角函数
Cos	余弦三角函数
Exp	指数函数
Frac	返回参数的小数部分
Int	返回参数的整数部分
Ln	计算自然对数
Pi	返回圆周率（3.14159265358979323 85）
Sin	正弦三角函数
Sqr	求平方函数
Sqrt	求平方根

1.2.2.2 控制台例程

RTL 提供的控制台例程如表 1-4 所示，这些例程可用来编写界面类似于 DOS 程序的 Windows 应用。

表 1-4 控制台例程

名 称	说 明
ClrEol	清出光标所在行光标后面的所有字符
ClrScr	清出屏幕内容
CursorTo	在虚拟屏幕中移动光标
DoneWinCrt	关闭一个 CRT 窗口
GotoXY	在虚拟屏幕中移动光标位置
InitWinCrt	建立一个 CRT 窗口
KeyPressed	测试是否有键盘按下
ReadBuf	在 CRT 窗口中输入一行
ReadKey	从键盘中读取字符
ScrollTo	滚动 CRT 窗口
TrackCursor	滚动窗口，使得光标可见
WhereX	返回光标的 X 坐标
WhereY	返回光标的 Y 坐标
WriteBuf	将字一批符写入 CRT 窗口
WriteChar	向 CRT 窗口写入单个字符

1.2.2.3 日期/时间例程

RTL 提供的日期/时间例程如表 1-5 所示。

表 1-5 日期/时间例程

名 称	说 明
Date	返回当前时间
DateTimeToStr	将日期时间值转换成字符串
DateTimeToString	使用给定的格式化串将日期时间数值转换成字符串
DateToStr	使用 ShortDateFormat 全局变量将日期时间值转换成字符串
DayOfWeek	DayOfWeek 返回 1-7 之间的数值表示给定日期为星期几
DecodeDate	过程将 Date 参数分解成年、月、日值
DecodeTime	过程将 Date 参数分解成时、分、秒值
EncodeDate	编码一个日期值
EncodeTime	编码一个时间值
FormatDateTime	用给定的格式串格式化日期时间值
Now	Now 返回当前日期和时间
StrToDate	将字符串转换成日期格式
StrToDateTime	将字符串转换成日期时间格式
StrToTime	字符串转换成时间格式
Time	返回当前时间
TimeToStr	将时间转换成字符串

1.2.2.4 动态内存申请例程

RTL 提供的动态内存申请例程如表 1-6 所示。

表 1-6 动态内存申请例程

名称	说明
Dispose	释放 P 指向的内存块
FreeMem	释放给定尺寸的动态变量
GetMem	申请内存空间
New	申请动态变量
MaxAvail	返回堆中最大可用的连续空间
MemAvail	返回堆中的自由空间数量

1.2.2.5 文件管理例程

RTL 提供的文件管理例程如表 1-7 所示。

表 1-7 文件管理例程

名 称	说 明
ChangeFileExt	修改文件扩展名称
DateTimeToFileDate	将 Delphi 日期格式转换成为 DOS 日期格式
DeleteFile	删除文件
DiskFree	返回磁盘剩余空间
DiskSize	返回磁盘空间
ExpandFileName	返回完整的路径字符串
ExtractFileExt	返回文件扩展名称
ExtractFileName	返回文件名称
ExtractFilePath	返回文件的路径字符串
FileAge	返回文件日期和时间
FileCreate	建立文件
FileClose	关闭文件
FileDateToDateTime	将 DOS 格式的日期转换成 Delphi 日期
FileExists	测试文件是否存在
FileGetAttr	返回文件属性
FileGetDate	返回文件的 DOS 格式的日期和时间
FileRead	读文件
FileSearch	在目录中查找文件
FileSeek	文件定位（移动文件读写指针）
FileSetAttr	设置文件属性
FileSetDate	设置文件的日期和时间
FileOpen	打开文件
FileWrite	文件写入
FindClose	关闭 FindFirst/FindNext 调用
FindFirst	查找文件
FindNext	查找下一个匹配的文件
RenameFile	文件换名

1.2.2.6 浮点转换例程

RTL 提供的浮点转换例程如表 1-8 所示。

表 1-8 浮点转换例程

名 称	说 明
FloatToDecimal	将浮点数转换成十进制表示格式
FloatToStrF	将浮点数转换成字符串格式
FloatToStr	将浮点数转换成字符串格式
FloatToText	将浮点值转换成十进制格式
FloatToTextFmt	将浮点值转换成十进制格式
FormatFloat	用给定的格式化字符串格式化浮点值
StrToFloat	将字符串转换成浮点值
TextToFloat	将以#0结束的字符串转换成浮点值

1.2.2.7 流程控制例程

RTL 提供的流程控制例程如表 1-9 所示。

表 1-9 流程控制例程

名 称	说 明
Break	终止 for、while、repeat 语句的执行
Continue	跳过后续语句，执行新一轮循环 (在 for、while 或 repeat 语句中)
Exit	立即退出本过程
Halt	非正常结束程序执行
RunError	激发一个运行时间错误，并结束程序执行

1.2.2.8 系统 I/O 例程

RTL 提供的系统 I/O 例程如表 1-10 所示。

表 1-10 系统 I/O 例程

名 称	说 明
AssignFile	将一个文件字符串与文件变量关联
CloseFile	关闭一个文件（文件变量不再与磁盘文件相关联）
Eof	测试文件指针是否指向文件结束位置
Erase	删除与文件变量关联的磁盘文件。
FilePos	返回文件指针的当前位置
FileSize	返回文件以字节为单位或记录为单位的长度
GetDir	返回磁盘的当前目录
IOResult	返回上一次 I/O 操作后的状态值
MkDir	建立一个子目录
Rename	修改文件名称
Reset	打开一个存在的文件
Rewrite	创建一个文件并打开文件
RmDir	删除一个子目录，目录必须为空
Seek	移动文件指针
Truncate	从当前位置截断文件

1.2.2.9 内存管理例程

RTL 提供的内存管理例程如表 1-11 所示。

表 1-11 内存管理例程

名 称	说 明
AllocMem	在堆中申请一块空间
ReAllocMem	重新申请空间（扩充或压缩内存块）

1.2.2.10 杂用例程

RTL 提供的杂用例程如表 1-12 所示。

表 1-12 杂用例程

名 称	说 明
AddExitProc	将一个过程添加到系统运行时间库的结束例程列表中
ChDir	改变当前目录
Exclude	从集合中删除一个集合元素
FillChar	在指定的地址处填充给定长度的字节（该函数不做范围检查）
Hi	返回高字节值
Include	给集合添加元素
Lo	返回变量的低字节值
Move	在源和目的之间进行字节复制
ParamCount	返回传递给程序的命令行中的参数个数
ParamStr	返回特定的命令行参数
Random	返回一个随机数
Randomize	初始化系统内建立的随机数发生器
SizeOf	返回参数的尺寸
Swap	交换高低字节（用于 Integer 或 WORD 类型）
UpCase	将字符转换成大写

1.2.2.11 序数类例程

RTL 提供的序数类例程如表 1-13 所示。

表 1-13 序数类例程

名 称	说 明
Dec	将参数 X 减去 1 或 N
Inc	将参数 X 增加 1 或 N
Odd	测试 X 是否为奇数
Pred	返回 X 前面的一个值
Succ	返回 X 的后一个值

1.2.2.12 指针与地址例程

RTL 提供的指针与地址例程如表 1-14 所示。

表 1-14 指针与地址例程

名 称	说明
Addr	返回对象的地址
Assigned	测试函数或过程变量是否为空
CSeg	返回当前 CS 寄存器的值
DSeg	返回当前 DS 寄存器的值
Ofs	返回给定对象的偏移
Ptr	将段地址和偏移转换成指针
Seg	返回给定对象的段地址
SPtr	返回 SP 寄存器的值
SSeg	返回 SS 寄存器的值

1.2.2.13 字符串格式化例程

RTL 提供的字符串格式化例程如表 1-15 所示。

表 1-15 字符串格式化例程

名 称	说 明
FmtStr	将一系列变元格式化成字符串
Format	格式化一系列变元，返回 Pascal 格式的字符串
FormatBuf	格式化一系列变元
StrFmt	格式化一系列变元
StrLFmt	格式化一系列变元，返回一个指向结果缓冲区的指针

1.2.2.14 Pascal 字符串操作例程

RTL 提供的 Pascal 字符串操作例程如表 1-16 所示。

表 1-16 Pascal 字符串操作例程

名 称	说 明
AnsiCompareStr	比较字符串（大小写敏感）
AnsiCompareText	比较字符串（大小写不敏感）
AnsiLowerCase	将字符串转换成小写形式
AnsiUpperCase	将字符串转换成大写形式
AppendStr	将字符串添加到另一个字符串之后
AssignStr	为字符串 PString 申请空间
CompareStr	比较字符串（大小写敏感）
CompareText	比较字符串（大小不写敏感）
Concat	合并一个或多个字符串
Copy	返回字符串的子串
Delete	在字符串中删除子串
DisposeStr	释放字符串
FmtLoadStr	从程序字符串资源表中装入字符串
Insert	将字符串作为子串插入一个字符串中
IntToHex	将整数转换成十六进制
IntToStr	将整数转换成字符串
IsValidIdent	测试字符串是否为合法的标识符
Length	求字符串的长度
LoadStr	从执行文件的字符串资源中装入字符串
LowerCase	将字符串转换成小写形式
NewStr	在堆空间中申请一个新字符串
Pos	在字符串中查找字符串
Str	将数字值转换成字符串
StrToInt	将字符串转换成整数
StrToIntDef	将字符串转换成整数，或返回缺省值
UpperCase	将字符串转换成大写形式
Val	将字符串转换成数字形式

1.2.2.15 零结束字符串操作例程

RTL 提供的零结束字符串操作例程如表 1-17 所示。

表 1-17 零结束字符串操作例程

名 称	说 明
StrAlloc	申请字符串缓冲区
StrBufSize	计算字符串缓冲区中可存放的字符总数
StrCat	字符串拼接
StrComp	比较两个字符串
StrCopy	复制字符串
StrDispose	释放字符串缓冲区
StrECopy	字符串复制，返回指向字符串结尾的指针
StrEnd	返回指向字符串结尾的指针
StrLCat	在字符串尾部添加字符
StrIComp	比较两个字符串，不区分大小写
StrLComp	比较字符串中给定个数的字符
StrLCopy	复制字符串中给定个数的字符
StrLen	返回字符串的长度（字符个数）
StrLIComp	比较字符串中给定个数的字符，不区分大小写
StrLower	将字符串转换成小写
StrMove	复制字符串中给定个数的字符
StrNew	在堆中申请字符串
StrPas	将以 null 结尾的字符串转换成 Pascal 格式的字符串
StrPCopy	将 Pascal 格式字符串复制到 null 结尾的字符串中
StrPLCopy	将指定个数的字符从 Pascal 字符串复制到 0 结尾的字符串中
StrPos	返回子串在字符串中出现的位置指针
StrScan	返回字符在字符串中首次出现的位置指针
StrRScan	返回字符在字符串中最后出现的位置指针
StrUpper	将字符串转换成大写形式

1.2.2.16 文本文件操作例程

RTL 提供的文本文件操作例程如表 1-18 所示。

表 1-18 文本文件操作例程

名 称	说 明
Append	以添加的方式打开现有文件
Eoln	测试文件指针是否在行尾
Flush	将文本文件缓冲区中的内容写入磁盘
Read	对于有类型文件，读入元素到程序变量中 对于文本文件读入一个或多个值
Readln	读入一行，并将文件定位到下一行
SeekEof	返回文件结束状态
SeekEoln	返回文件行结束状态
SetTextBuf	给文本文件定义一个 I/O 缓冲区
Write	对于类型文件，将变量写入文件单元中 对于文本文件，将多个值写入文件中
Writeln	该过程用于文本文件，完成 Write 的功能，同时向文件写入一个行结束符

1.2.2.17 转换例程

RTL 提供的转换例程如表 1-19 所示。

表 1-19 转换例程

名 称	
Chr	将 ASCII 值转换成字符
High	返回参数范围的最大值
Low	返回参数范围的最小值
Ord	返回参数的位置序数值
Round	将实数类型的数据舍入
Trunc	将一个 real 值截取成 Integer 值

1.2.2.18 无类型文件处理例程

RTL 提供的无类型文件处理例程如表 1-20 所示。

表 1-20 无类型文件处理例程

名 称	说 明
BlockRead	将一个或多个记录读入变量中
BlockWrite	将变量中的一个或多个记录写入文件

1.3 VCL简介

前面一节中，我们罗列了 Delphi 运行时间库 RTL 中的所有内容，有关 Delphi RTL 中函数、过程的细节和使用方法，将在后面一章中讲解。

在本节中，我们将从总体上讲述 Delphi VCL 库的组织和结构，介绍 Delphi 可视组件库 VCL 的组成、继承关系和 VCL 库包含的各种内容，如组件、对象、函数和过程等。

1.3.1 VCL祖先类层次

Delphi 可视组件库 VCL 是一个完全的面向对象的程序库，具有严格的层次结构关系，VCL 中大量的组件、对象都是从一些最基本的对象继承而来。VCL 的最基本的层次结构，如图 1-3 所示。

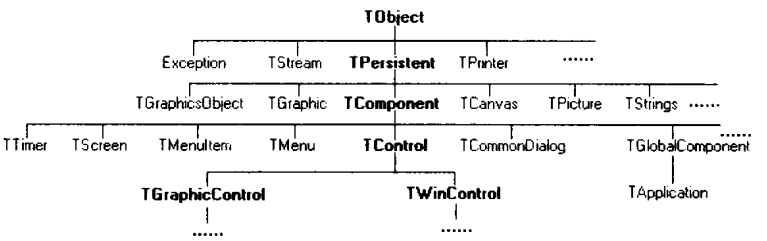

图1-3 Delphi可视组件库VCL祖先类继承关系

Delphi 可视组件库 VCL 定义了许多类和对象， VCL 可视组件库中的所有各组件和控件(Controls)从六个关键的类继承而来，Delphi IDE 主窗口中各个组件页中罗列的所有组件都是这六个类的后代。

在图 1-3 中，TObject、TPersistent、TComponent、TControl、TGraphicControl 和 TWinControl 构成了五级类层次，每一级定义了各自的功能特性，它们是 VCL 的祖先类，VCL 的各组件和对象继承了这几个类的功能和特性，了解这几个类，可以理解 VCL 组件和对象的绝大部分功能和用途。

1.3.1.1 TObject 与 TPersistent 对象

如图 1-3 所示，TObject 是 VCL 的基类，VCL 中所有的对象和组件都从该类继承而来，事实上，TObject 是 Delphi 系统中所有类的基类。

TObject 提供了一些基础的方法，VCL 库中的所有对象和组件都继承这些方法，例如用于给对象副本（对象实例，Instance）申请和释放空间的 Create 和 Free 方法。

TObject 还具有许多用于描述对象的类型、类名称以及父类的有关信息的方法，TObject 包含的各种方法如图 1 - 4 所示。

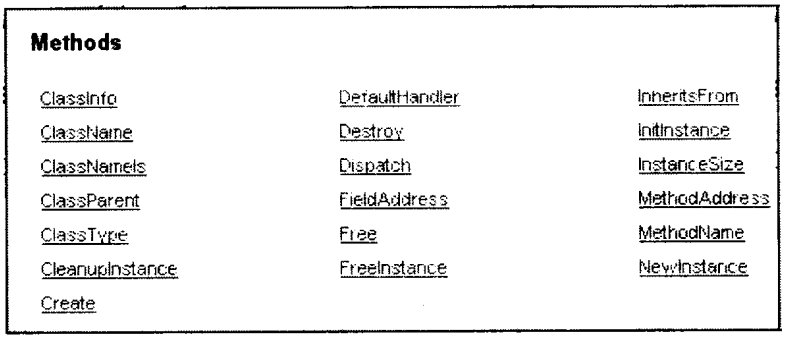

图1-4 TObject对象方法

TObject 对象申明了多个方法，这些方法被所有的对象（类）继承和重载，这些方法主要完成如下一些功能：

(1) 构造和析构对象变量

在 TObject 中，与对象变量构造和析构相关的方法包括 Create、NewInstance、Destroy、Free、FreeInstance、CleanupInstance。

Create 与 Destroy 是构造和析构函数，Free 是更安全的调用析构函数的形式，NewInstance、FreeInstance、CleanupInstance 是用于对象存储空间的申请释放和初始化的方法。

(2) 类型信息相关方法（Type-information methods）

在 TObject 对象中，ClassInfo、ClassName、ClassNameIs、ClassParent、ClassType、InheritsFrom 以及 InstanceSize 方法提供了对象类型以及对象实例的各种有用信息。另外，我们还可以通过 FieldAddress、MethodAddress 和 MethodName 方法获取具有 Published 特性的成员的运行时间类型信息（run-time type information，RTTI）。

(3) 用于消息处理的方法（Message-handling methods）

对象的 Dispatch 和 DefaultHandler 可用于分发、处理系统消息。

TPersistent 是 TObject 的直接后代，该对象实现了从流式文件中读取对象属性和将属性写入文件的方法。TPersistent 除了继承了 TObject 的方法外，新定义了三个方法。AssignTo、

DefineProperties、Assign。

1.3.1.2 TComponent 和 TControl 组件

从图 1-4 中可以看出，TComponent 直接从 TPersistent 对象继承而来，是 Delphi VCL 中所有组件的祖先。

TComponent 定义了组件的各种基本操作和属性，实现了组件的共有的功能，比如可以被放置在系统组件栏上的能力、可以在窗体设计中被直接操作的能力、可以包含其它组件的能力等，除继承了祖先对象的各方法外，TComponent 组件定义了自己的属性和方法，如图 1-5 和图 1-6 所示。

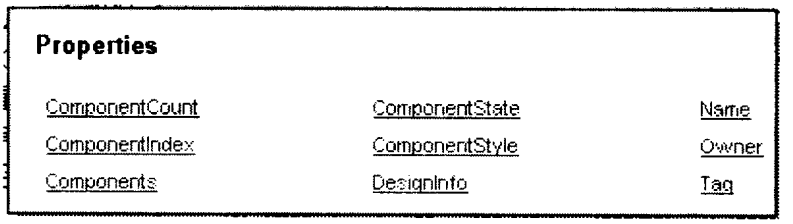

图1-5 TComponent 组件定义的属性

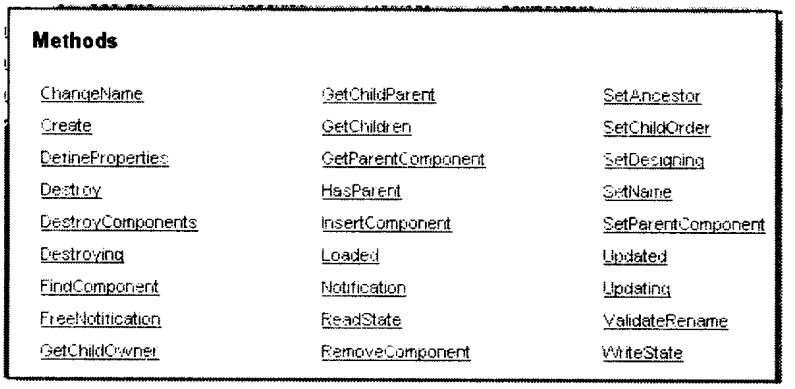

图1-6 TComponent 组件定义的方法（含继承的方法）

TComponent 定义的属性和方法几乎都是 public 或 published 成员，但也具有一些 protected 方法，TComponent 组件的后代可以继承或重载这些方法。

TComponent 新定义的方法主要用于实现如下一些功能：

(1) 操作 Name 属性的方法

SetName 方法是一个虚拟方法，用于 Name 属性的写入。Name 属性调用了 ChangeName 方法，该方法完成真正的修改名。

(2) 用于文件和流操作的方法

用于文件和流操作的相关方法包括 ReadState、WriteState、 HasParent、Loaded 等。ReadState 和 WriteState 方法用于读和写组件的属性值（形成文件对象），HasParent 方法确定组件是否具有“父亲”组件来处理自己的存储。Loaded 方法用于组件从文件中装入后的初始化。另外 TComponent 还重载了 DefineProperties 方法。

(3) 用于管理属于本组件的其它组件的方法

组件的 ValidateRename 方法保证修改一个属于本组件的组件的名称不会引起名字冲突。WriteComponents 将所属的组件写入到文件，形成文件对象。Notification 方法可以给所属的方法发送消息，将消息通知所属的所有组件。SetDesigning 方法用于设置组件的设计状态标志（design-mode flag）。

虽然 TComponent 是所有组件的缺省祖先，但从 TComponent 组件下，只能直接派生非可视组件，所有的可视组件是从 TComponent 的直接下级 TControl 继承而来的，TControl 是一个控件类组件。一个控件在程序运行时是可见的，我们使用的大多数组件都是从 TControl 继承下来的可视组件。当然，我们也使用直接从 TComponent 继承的非可视组件，例如 TMenu、TMainMenu、TPopup 等。

TControl 定义了各种控件所共有的基本属性、事件和方法，如图 1-7、图 1-8 和图 1-9 所示。

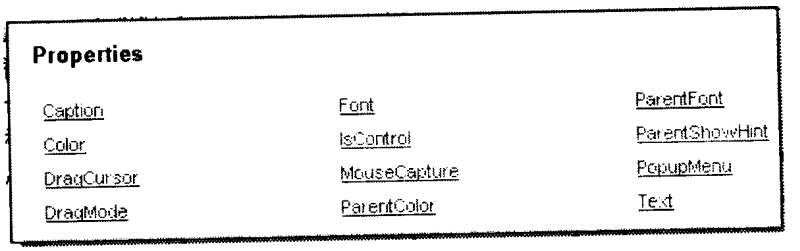

图1-7 TControl组件的属性

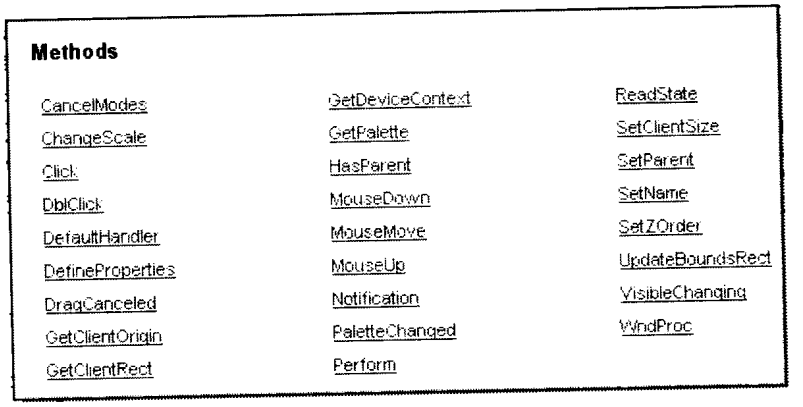

图1-8 TControl组件的方法

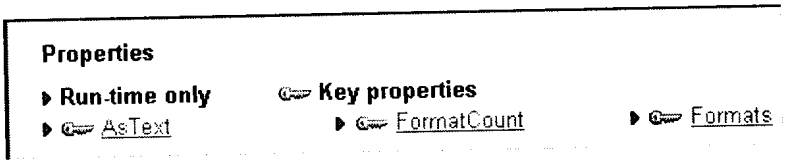

图1-9 TControl组件的事件

TControl 组件是一个抽象的可视组件，可视组件指在程序运行时可以看见的或可以进行交互操作的组件，该组件定义了许多可视性属性，如色彩、字体、标题等，还具有处理用户鼠标交互操作的方法和事件，如 Click、DblClick 方法以及 OnClick、OnDblClick、OnMouseDown 等事件，参见图 1-8 和图 1-9。

在 TControl 组件下面，是两个可视组件 TWinControl 与 TGraphicControl，这两个组件分别实现了两类具有不同功能的可视组件。

1.3.1.3　TWinControl 与 TGraphicControl

TWinControl 与 TGraphicControl 是 VCL 类层次中的最后一个抽象层次，了解 TWinControl 与 TGraphicControl 之间的异同，对我们理解 VCL 中的各种组件非常重要。

正如这两个组件的名称所表示的一样，TWinControl 是所有具有窗口特征的控件（窗口类组件）的祖先，包含 Windows 系统的标准控件如编辑框、列表框、按钮等。窗口类控件（组件）具有三个基本特征，这就是：

(1) 可以接受输入焦点。在程序运行时，窗口类组件可以接收输入焦点（Focus），用户可以通过键盘操作它。

(2) 具有标准的窗口句柄。 该句柄由组件的 Handle 属性定义，并且所有的窗口类组件都是从 TWinControl 继承而来。

(3) 可以包含其它的组件（控件）。

在 VCL 库中，有大量的组件是从 TWinControl 继承而来的，TWinControl 是用户可以交互操作的各组件的祖先，TWinControl 定义了许多用于实现上述功能的属性、事件和方法，如图 1-10、图 1-11 和图 1-12 所示。

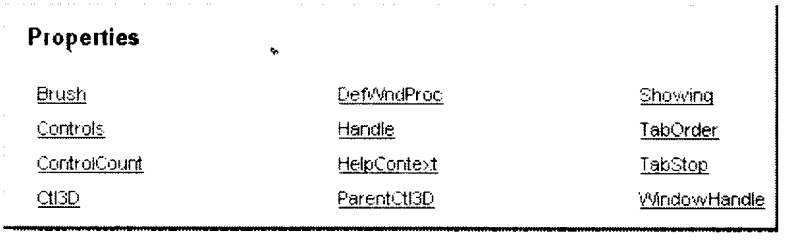

图1-10　TWinControl组件的属性

与 TWinControl 不同 TGraphicControl 没有 Handle 属性，也就是没有窗口句柄，不能接收窗口输入焦点，也不能包含其它的组件，但是，这种非窗口类组件有一个最大的优点就是不使用 Windows 资源，从而节省系统资源，并能使程序运行得更快。所以，在不需要与程序人员交互的地方应尽量使用非 Windows 类组件。

Delphi 支持通过 TGraphicControl 建立自己的组件，TGraphicControl 是所有非 Windows 类组件的祖先，是一个抽象组件，该组件除继承了 TControl 的属性、方法和事件外，另外定义了一个 Canvas 属性（该属性可用于绘制），定义了一个 Paint 方法，可以处理 WM_PAINT 消息。

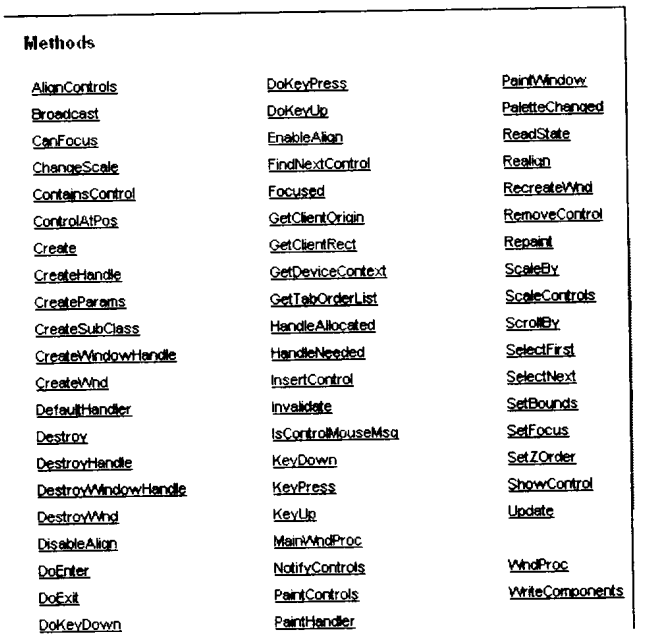

图1-11 TWinControl组件的方法

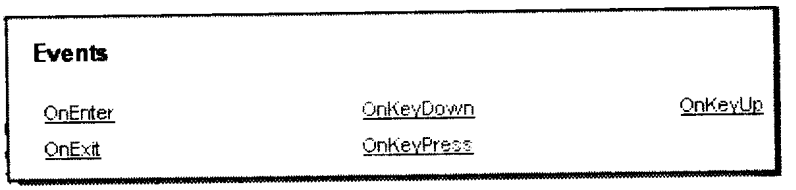

图1-12 TWinControl组件的事件

1.3.2 VCL类库的组件和对象

从总体上看，VCL 库包含基础对象和组件两大部分，对象是 VCL 库的基础，事实上，组件也是从对象继承而来的，基础对象给 VCL 库提供各种基本功能。

组件又分为可视组件和非可视组件两大部分，其中，可视组件又称为控件，它们在程序执行阶段是可见的；另外，可视组件又分为窗口类组件和非窗口类组件，VCL 库分类划分如图 1-1 所示。在图 1-1 中，几个组成部分包含的具体的对象如下：

(1) 基础对象

VCL 库中包含各类用途的对象（Object），这些对象是从 TObject、TPersistent 继承而来，这些对象提供了各种基础功能，是构筑 VCL 库的基础。系统包含的主要基础对象如下：

TBitmap	TGraphic	ToutlineNode
TBlobStream	TGraphicsObject	TParam
TBrush	TIcon	TParams
TCanvas	TIndexDef	TPen

TClipboard	TIndexDefs	TPicture
TcontrolScrollBar	TIniFile	TPrinter
TFieldDef	TList	TStringList
TFieldDefs	TMetafile	TStrings
TFont	TOLEDropNotify	

(2) 组件

组件是 VCL 库功能强大的具体体现，是 Delphi 应用程序的基本模块，通过将组件选择到窗体中，设置适当的属性，编写一定的事件处理程序，就可以快速构造一个应用程序。Delphi 组件库中提供的众多组件如下：

TApplication	TAutoIncField	TBatchMove	TBCDField
TBevel	TBitBtn	TBlobField	TBooleanField
TButton	BytesField	TCheckBox	TColorDialog
TComboBox	TCurrencyField	TDatabase	TDataModule
TDataSource	TDateField	TDateTimeField	TDBCheckBox
TDBComboBox	TDBCtrlGrid	TDBCtrlPanel	TDBEdit
TDBGrid	TDBImage	TDBListBox	TDBLookupCombo
TDBLookupComboBox		TDBLookupList	TDBLookupListBox
TDBMemo	TDBNavigator	TDBRadioGroup	TDBText
TDDEClientConv	TDDEClientItem	TDDEServerConv	TDDEServerItem
TdirectoryListBox	TDrawGrid	TDriveComboBox	TEdit
TField	TFileListBox	TFilterComboBox	TFindDialog
TFloatField	TFontDialog	TForm	TGraphicField
TGroupBox	THeader	THeaderControl	THotKey
TImage	TImageList	TIntegerField	TLabel
TListBox	TListView	TMainMenu	TMaskEdit
TMediaPlayer	TMemo	TMemoField	TMenuItem
TNotebook	TOLEContainer	TOpenDialog	TOutline
TOutline	TPageControl	TPaintBox	TPanel
TPopupMenu	TPrintDialog	TPrinterSetupDialog	
TProgressBar	TQuery	TRadioButton	TRadioGroup
TReplaceDialog	TReport	TRichEdit	TSaveDialog
TScreen	TScrollBar	TScrollBox	TSession
TShape	TSmallIntField	TSpeedButton	TStatusBar
TStoredProc	TStringField	TStrringGrid	TTabbedNotebook
TTabControl	TTable	TTabSet	TTabSheet
TTimeField	TTable	TTimer	TTrackBar
TTreeView	TUpdateSQL	TUpDown	TVarBytesField
TWordField			

上面的各组件，都是从 TComponent 继承而来的，其中的大部分组件在 IDE 的组件栏中可以看到，也就是说，大部分组件都是可视组件，是从 TComponent 的下级组件 TControl 继