用 Visual Basic 开发 DOS 程序的 Windows 用户界面

冯惠军　（石家庄铁道学院）

摘要：本文介绍利用 Visual Basic 为 DOS 应用程序开发 Windows 用户界面的一些技巧和应用经验。即如何建立 DOS 应用程序的 Windows 用户界面、如何在 Windows 程序中运行 DOS 程序和在后台运行 DOS 程序、怎样监测 DOS 程序的执行、如何构造 DOS 程序的命令行参数和反馈 DOS 程序的提示信息。

关键词：Visual Basic　Windows　用户界面　DOS 程序　后台运行

Windows 操作系统为微机用户提供了一个直观的、图形丰富的工作环境，Windows 应用程序提供的图形用户界面使应用程序更易于学习和使用，用户只要简单地用鼠标点击一下菜单或图标就可执行工作，而不必键入较长的命令，使人机交互更为方便。尽管 Windows 应用程序为我们带来了许多好处，但对一些具有完整功能、经常使用的 DOS 应用程序（例如文件压缩程序 ARJ、PKZIP 等）和专业性比较强的程序（如 SIMAN 仿真语言）目前还不可能完全放弃不用。虽然有些 DOS 程序功能强大，但是命令行操作和参数匹配极为复杂，缺乏 Windows 程序具有的用户友好性这一优点。尽管用户可以自己动手或等待开发商为一个 DOS 应用程序重新编写具有相同功能的 Windows 版本，但一般情况下是不现实的。一个行之有效的方法是为 DOS 应用程序开发一个基于 Windows 的用户界面，这个界面体现了 DOS 程序的功能。虽然在执行时仍然需要从 Windows 到 DOS 的切换，但通过简单技术可以把这一事实隐藏起来，使用户根本觉察不出这个基于 Windows 的用户界面实际上是一个基于 DOS 程序的命令行驱动程序。因此对用户来说，这个程序和一个真正的 Windows 应用程序并没有什么区别。笔者根据编程经验，介绍利用 Visual Basic 开发 DOS 应用程序的 Windows 用户界面的技术技巧。

1. 建立 DOS 应用程序的 Windows 用户界面

为了在 Windows 程序中实现 DOS 程序的功能，首先要设计一个体现 DOS 程序的 Windows 用户界面。在 VB 开发环境下，利用窗体（Form）或多文档界面（MDI）作为整个 DOS 程序的 Windows 用户界面。可以利用菜单和命令按钮实现对 DOS 程序的调用，利用文本框接受用户输入，利用标签、文本框或列表框实现向用户显示提示信息，利用单选钮和复选钮实现 DOS 命令行参数的选择和组合。Windows 界面的具体布局和设置根据开发人员自己的喜爱和程序类型来设计。总之利用 VB 提供的控件，可以很方便地为一个 DOS 应用程序设计一个 Windows 用户界面。

2. 在 Windows 程序中启动 DOS 程序

在 Windows 应用程序中启动一个 DOS 程序可以采用两种方法，一是利用 VB 提供的 Shell（函数），一种是利用 Windows 提供的应用程序接口（API）函数 WinExec()。

Shell()函数的格式为：

Shell(命令字符串,窗口类型)

命令字符串是要执行的应用程序的命令行，包括程序的扩展名和命令行参数。Shell()只限于执行扩展名为 COM、EXE、BAT 或 PIF 的类型的程序，否则会产生错误。

窗口类型是一个整型值，指明程序运行时所在的窗口类型。由于 Windows 程序是在一个窗口中执行的，即使执行 DOS 程序也是在一个窗口（DOS 窗口）中执行的。窗口类型指明运行程序的窗口是最大化还是最小化，具有焦点还是不具有焦点。

如果函数调用一个 DOS 应用程序运行成功，它返回一个启动程序的任务标识（ID），这个标识是表示正在执行的程序的唯一标识。若调用失败，VB 会给出一个出错信息。

例如，在 Windows 程序中调用 ARJ 程序压缩当前目录下的所有文件，命令形式为：

X = Shell(":C:\ABC\ARJ.EXE A Sample",7)

注意：命令字符串包括了路径名、扩展名和命令行参数，窗口类型值为 7，表示 DOS 程序在一个不具有焦点的最小化窗口中执行。

WinExec()函数，必须在 VB 程序中作以下声明：

Declare Function WinExec Lib"Kernel"(ByVallpCmdLine As String ,ByVal nCmdShow As Integer) As Integer

例如，上述例子中用 WinExec()函数调用 ARJ 程序的命令形式为：

X = WinExec("C:\ABC\ARJ.EXE A Sample",0)

注意：命令字符串与 Shell()函数完全相同，窗口类型值为 0，表示指定 DOS 程序在一个隐含的窗口中执行。

3. 在后台运行 DOS 程序

当用 Shell()函数或 WinExec()函数启动一个 DOS 程序后，几乎立即返回 Windows 应用程序，然后将切换到 DOS 环境中运行这个 DOS 程序，这就破坏了 Windows 用户界面的

完整性。虽然可以指定 Shell()函数的窗口类型参数为 7,使 DOS 程序在一个不具有焦点的最小化窗口中运行,或指定 WinExec()函数的窗口类型参数为 0,使 DOS 程序在一个隐含的窗口中运行。但是使用 Shell()函数必须切换到 DOS 窗口使之具有焦点才能执行,使用 WinExec()函数还是自动切换到 DOS 环境中运行。

如何使启动的 DOS 程序在后台运行,而保持 Windows 应用程序在前台运行?根据笔者的开发经验,为使用的 DOS 程序建立相应的 PIF 文件就可以有效地控制 DOS 程序的运行,满足我们的目标。PIF 文件是 Windows 的可执行文件,它提供了要求 Windows 如何运行 DOS 程序的信息。PIF 文件编辑器有许多选项,主要填以下信息:要执行的 DOS 程序名(包含扩展名)、DOS 程序启动目录、后台执行、在窗口中执行、退出时关闭窗口,然后选择高级选项,设置后台优先权为 5000。这样在 Shell()或 WinExec()函数调用 DOS 程序的 PIF 文件就可以使启动的 DOS 程序在后台运行,而保持 Windows 用户界面不变,使用户觉察不出正在执行 DOS 程序。

4.监测 DOS 程序的执行

无论使用 Shell()函数还是 WinExec()函数,在启动了指定的 DOS 程序之后都会立即返回 Windows 程序,它们都没有提供使 Windows 程序暂停运行等到启动的 DOS 程序运行结束的内部选项。

如何才能知道启动的 DOS 程序是否已经结束?利用 Windows 提供的 GetModuleUsage()函数可以实现这一目标。利用 Shell()或 WinExec 函数返回的已启动程序的任务标识(ID)调用 GetModuleUsage()函数即可判断启动的 DOS 程序是否正在运行。如果程序正在运行,GetModuleUsage()函数返回 True,否则返回 False,通过在一个循环中调用 GetModuleUsage()函数使 Windows 程序一直处于等待状态,直到 GetModuleUsage()返回 False 时才退出循环继续运行,这样就可以使 Windows 程序一直等到启动的 DOS 程序运行结束,使用 GetModuleUsage()函数,必须在 VB 程序中作以下声明:

```
Declare Function GetModuleUsage Lib "Kernel" (ByVal hModule As Integer) As Integer
```

以上过程使用 Shell()函数的实现程序如下:

```
Temp% = Shell("C:\ABC\ARJ.EXE A Sample",7)
If Temp% < 32 Then
  MsgBox"命令错误! 不能执行",48,"程序运行"
  Exit Sub
End If
While GetModuleUsage(Temp%)
  Wait% = DoEvents()
Wend
```

在上面的程序中,While 循环中的 DoEvents()调用为 ARJ.EXE 提供了运行机会。这个循环不断执行,直到 GetModuleUsage()返回 False,表明 Shell()启动的 DOS 程序运行结束,然后退出循环继续运行 Windows 程序。

5.构造命令行参数

许多 DOS 应用程序都支持相当多的命令行选项和开关,这些选项为用户提供了大量的选择。例如 ARJ 程序支持多达几十种命令选项,可以满足用户不同的功能需要,但这些选项常常使用户感到眼花缭乱。在 Windows 用户界面中,就可以实现让用户用鼠标对这些选项进行直观的选择和组合,这比在 DOS 系统键入一长串命令要容易和简单的多。

在 Windows 程序中就可根据用户的这种选择和输入构造 DOS 程序的命令行参数,利用 VB 很容易实现这种功能。可以利用文本框或列表框接受用户输入,利用单选钮或复选钮实现命令行参数的选择和组合,然后根据用户的输入和选择构造一个命令字符串,把这个字符串传递给 Shell()或 WinExec()函数就可以实现用户选择的功能。例如,用 ARJ 程序把 C:\ABC 子目录中的所有文件以及所有子目录都压缩到一个名为 Sample 的文件中,对压缩的文件进行验证,并把这个文件放在 C:\DEF 子目录中。

根据以上选择可以构造一个命令字符串 S$:

```
S$ = "C:\ABC\ARJ.PIF A -R -JT C:\DEF\Sample C:\ABC\*.*"
```

然后把这个字符串传递给 Shell()函数:

```
Temp% = Shell(S$,7)
```

6.反馈 DOS 程序的提示信息

有些 DOS 程序在运行过程中,会在屏幕上向用户显示一些提示信息。由于 DOS 程序是在一个隐含的窗口中运行的,因而用户看不到这些显示信息。即使程序运行出现错误,用户也无法知道。因此,需要采用一些方法把 DOS 程序的提示信息显示在 Windows 窗口中,似乎只能利用文件传递的方法实现这一功能。利用 DOS 的重定向功能(>)把 DOS 程序显示在屏幕上的提示信息定向输出到一个指定的文件中,然后在 Windows 程序中重新打开这个文件,并把这个文件的内容显示在 Windows 窗口中。

但在实现过程中我们发现,无论使用 Shell()还是 WinExec()从 Windows 程序启动的 DOS 的.EXE 程序都不能被重定向,但如果利用 Shell()或 WinExec()启动一个批命令(.BAT)文件,在这个批命令文件中再启动 DOS 程序就可以实现输出的重定向。

以上介绍了利用 Visual Basic 为 DOS 应用程序开发 Windows 用户界面的一些技术技巧和应用经验,读者可以根据本文介绍的技巧为你的 DOS 程序开发友好的 Windows 界面。