在 VB 应用程序中实现多背景动画

李龙星　（洛阳工业高等专科学校计算机系　471003）

摘要：本文主要介绍了在 VB 应用程序中实现在多背景上文字与图片动画的方法。

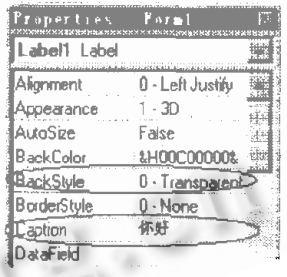
关键词：VB　动画　多背景

在 VB 应用程序中，可以通过定义图片控件在单背景下实现动画，这仅仅需要把图片控件的背景色设置成动画对象所在背景色即可。由于可见控件均是方形，简单的把这种方法移植到多背景下，画面上将会因出现不希望出现的颜色而破坏动画效果。为此，我们采用 Windows API 函数 BitBlt，利用它在搬动图片时的几种不同方式实现多背景图片动画。

1. 文本在多背景上的动画

首先把复杂图片（例如 windows 下的 leaves. bmp）放置在图片控件（Image）中，再把需要动画的文本（例如“你好”）放置在一个文本标签控件（不妨设其名字是 Label1）中。这里一定要把标签控件的属性 BackStyle 设置为 0. Transprent（透明）方式，默认方式为 1. Opaque（非透明）方式。设置时钟控件（不妨设其名字为 Timer1）的属性 Interval 为一合适的值，在其 Timer 事件的代码内填写如下程序段：

```
Private Sub Timer1-Timer()
    Label1.Move Label1.Left + 5, Label1.Top
End Sub
```



运行程序就会发现“你好”二字在水平移动，并且通过 image 控件中的复杂图片时，背景不会被破坏。可以通过动态改变 Label1 的字体属性，可以实现文本移动时字体大小与颜色的变化；也可以设置多个 Label 控件实现文本的飞入飞出等，以实现更为复杂的效果。

2. 在多背景上实现图片动画

（1）在一全局模块中声明常数和 BitBlt 函数：

```
Public Const SRCAND = &H8800C6
```

```
Public Const SRCCOPY = &HCC0020
Public Const SRCPAINT = &HEE0086
Public Const SRCINVERT = &H660046
```

```
Declare Function BitBlt Lib "gdi32" Alias "BitBlt"
(ByVal hDestDC As Long, ByVal x As Long, ByVal y As
Long, ByVal nWidth As Long, ByVal nHeight As Long,
ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc
As Long, ByVal dwRop As Long) As Long
```

(2)在窗口中放置一个 Picture 控件，名字为 Picture1。在此控件内放置多彩图片，不妨放置 windows 下的 setup.bmp。由于要使用 BitBlt 函数，此控件的 ScaleMode 属性应该设置为 3.Pixel。

(3)准备让一个汽车在图片上移动，不妨就取 windows 下的 cars.bmp。为此需要对此图片进行一番加工，具体步骤如下：

①运行 windows 画笔，打开位图 cars.bmp。利用工具把位图除汽车之外的其他部分全部涂成黑色，汽车部分不予改变。另存位图为 car1.bmp。

②再打开位图 cars.bmp，此次利用工具先把汽车部分涂黑，再把其余部分填成白色，另存位图为 car2.bmp。

(4)在窗口中放置三个 Picture 控件，分别命名为 Picture2，Picture3 和 Picture4，它们的 ScaleMode 属性均设置为 3.Pixel；AutoSize 均设置为 True；AutoReDraw 均设置为 True；Visible 均设置为 False。Picture2 的 Picture 取 Car1.bmp，Picture3 和 Picture4 的 Picture 均取 Car2.bmp。由于 Car1.bmp 和 Car2.bmp 两个位图尺寸大小一样，所以，这三个 Picture 控件的 Width 与 Height 也完全一样。

(5)在窗口的 Load 事件中填写如下程序：

```
Dim RetVal%
RetVal = BitBlt(Picture4.hDC, 0, 0, Picture4.Width,
Picture4.Height, Picture2.hDC, 0, 80, SRCCOPY) '程序一开始运行即保存背景被画部分于 Picture4 中，以便恢复背景用。
```

(6)在窗口中放置一个时钟控件，设置属性 Enabled 为 False，设置 Interval 值为 100，在其 Timer 事件的代码段填写如下程序段：

```
Static i
Dim RetVal%
i = i + 1
RetVal% = BitBlt(Picture1.hDC, i - 1, 80, Picture4.
Width, Picture4.Height, Picture4.hDC, 0, 0, SRCCOPY) '
把保存在 Picture4 中的背景部分恢复到背景中。
RetVal% = BitBlt(Picture4.hDC, 0, 0, Picture4.
Width, Picture4.Height, Picture1.hDC, i, 80, SRCCOPY)'
保存即将被画的背景部分于 Picture4 中以便下一次恢复背景用。
RetVal% = BitBlt(Picture1.hDC, i, 80, Picture3.
Width, Picture3.Height, Picture3.hDC, 0, 0, SRCAND)'
先用 Car2.bmp 与背景作与
RetVal% = BitBlt(Picture1.hDC, i, 80, Picture3.
Width, Picture3.Height, Picture2.hDC, 0, 0, SRC-
PAINT)'再用 Car1.bmp 与背景作异或。
```

(7)添加一个按钮控件用以启动时钟。运行程序，点按按钮，即可看到汽车在背景上移动，而周围背景不会被破坏。

3.**原理分析**

在程序一开始运行即首先保存了背景将要被画的部分。在每一个时钟事件中，总是先恢复背景，再保存将要被画的背景部分，以便下一次恢复背景。真正显示汽车的过程由后两个 BitBlt 函数来完成，其原理如下：

让 Car2.bmp 先与背景作与，由于 Car2.bmp 中汽车部分全部是黑色的，与背景作与的结果是背景上汽车部分全变为黑色，而 Car2.bmp 中汽车周围是白色的，作用结果是其周围的背景保留，此时 Picture1 中除了准备显示汽车的地方是一个黑洞外，别的地方没有任何改变，这就是在背景上先打洞，再用 Car1.bmp 与背景上刚刚被 Car2.bmp 作用过的部分作异或。由于 Car1.bmp 中的汽车部分是正常的，与黑洞异或自然是正常汽车，而 Car1.bmp 的汽车周围均已涂黑，与背景异或自然不改变背景。尽管汽车不是矩形，但移动起来并不破坏背景，达到多背景动画的目的。

（来稿时间：1998 年 10 月）