

图 片 应 用

3.1 编程指导（三）

3.1.1 Visual Basic 支持的图片格式

Visual Basic 直接支持的图片格式有 Bitmap 位图、icon 图标、metafile 元文件、GIF 格式和 JPEG 格式，分别以.bmp、.ico、.wmf、.gif 和.jpg（或.jpeg）为文件的扩展名。其中.bmp 位图文件没有对图像进行压缩，所以文件较大；而.jpg 文件采用了 JPEG 标准进行了压缩，大小相当于相同内容位图文件的 1/20（压缩率可以调整），但 JPEG 有损压缩，会使图片清晰度降低。GIF 文件采用了无损压缩方法，却不支持真彩色。图标文件.ico 可以用作应用程序图标（显示在资源管理器中和窗口左上角）、工具栏按钮图片和拖放操作时的图标等。一个图标文件中可以包括不同分辨率和颜色数的多个图标，常用的有 16×16、32×32 和 64×64 等几种分辨率。Windows 会根据情况使用图标文件中的不同图标。WMF 元文件是矢量图片，缩放不会导致失真，一般保存简单的几何图形。

另外，Visual Basic 还支持光标文件.cur，用来定义鼠标指针形状。光标文件不但定义了鼠标指定的形状，还定义了一个“热点”（Hot Spot），即鼠标真正指向的点。譬如，常用的箭头形状指针的热点位于其最“尖”的位置上。改变鼠标指针的热点需要专门的光标编辑软件（如 Borland Image Editor）。

一般的图片处理程序可以处理并保存 bmp、gif 和.jpg 格式的图片文件，而光标、图标和元文件需要专用的软件进行编辑和生成。本书的样例程序均提供了要用到的各种图片文件。

3.1.2 图片的使用

（1）窗体、图像和图片框控件。通过 Picture 属性可以指定图片，对于窗体和图片框来说，指定的图片是背景图片，总是位于对象的左上角，如果不希望这种效果，可以使用绘图方法 PaintPicture 将图片绘制到任意位置上。

给 Picture 属性赋值有两种方法，一种是在设计时通过属性窗口赋值，另一种是运行时赋值（如使用 LoadPicture 函数加载图片并赋值）。前者将图片保存在二进制窗体文件.frx 中，生成可执行文件时会将图片编译到可执行文件中，程序运行时不再需要原来的图片文

件。这种方法的缺点是可执行文件太大。第二种方法必须保证程序在运行时图片文件在指定位置，否则 LoadPicture 函数会出错。

(2) 命令按钮、单选框和复选框控件。这三类对象也可以显示图片，需要先将其 Style 属性设为 1 (Graphical)，然后为 Picture 属性赋值即可。其他方面与图像和图片框控件相同。只要选择一个内容恰当、大小合适的图片文件，创建一个“图片按钮”相当简单。

(3) LoadPicture 函数的使用。

```
LoadPicture([filename], [size], [colordepth],[x,y])
```

LoadPicture 函数在程序运行时加载由 filename 参数（字符串类型）指定的图片文件。如果加载的是图标文件（.ico）会用到 size、colordepth、x 和 y 参数。size 参数指定要加载的是图标文件中的哪个具体的图标，使用 colordepth 指定使用的颜色数（详见表 3.1 和表 3.2）。

表 3.1 size 参数的意义

常量	参数值	意义
vbLPSmall	0	系统小图标
vbLPLarge	1	系统大图标，由视频驱动程序决定
vbLPSmallShell	2	Shell 小图标，由“控制面板”中的“显示属性”对话框中的大小设置决定
vbLPLargeShell	3	Shell 小图标，由“控制面板”中的“显示属性”对话框中的大小设置决定
vbLPCustom	4	自定义大小，由 x 和 y 参数提供大小值

表 3.2 colordepth 参数的意义

常量	参数值	意义	常量	参数值	意义
vbLPDefault	0	自动选择颜色	vbLPVGAColor	2	16 色
vbLPMonochrome	1	2 色	vbLPColor	3	256 色

例如，下面的语句分别加载位图和图标文件。

```
Picture1.Picture = LoadPicture("f:\a.bmp")
Me.Icon = LoadPicture("f:\b.ico",1)
```

不带任何参数的 LoadPicture 函数返回一个“空图片”用来清除对象所显示的内容。

(4) 使用 StdPicture 数据类型进行图片管理。StdPicture 是 Visual Basic 的图片对象型数据类型，可以用来保存一个图片对象。

```
Dim pic As StdPicture           '定义 StdPicture 类型的对象型变量
Set pic = LoadPicture("f:\a.bmp") '使用 Set 语句为变量赋值
Picture1.Picture = pic          '将变量的值赋给 Picture 属性
```

如果一个程序要用到大量的图片，常用的办法是定义一个 StdPicture 类型的数组来管理这些图片，再赋值给其他对象来显示图片内容。

3.1.3 大量图片的管理方法

除了使用上述的 `stdPicture` 类型的数组之外，还可以使用以下方法对大量的图片进行管理。

(1) 使用多个 `Image` 图像控件、`PictureBox` 图片框控件或控件数组。有人使用多个图像控件来保存多个图片，然后将其设置为不可见，通过赋值的办法将这些图片由其他控件显示。其实这不是一个好办法，因为图像和图片框控件会占用大量的系统资源，如果不是为了同时显示多个图片，不宜采用这种方法。

(2) 使用 `PictureClip` (图片裁剪) 控件。`PictureClip` 是一个 ActiveX 控件，在“部件”对话框中选择“Microsoft Picture Clip Control”将其添加到工程中，其工具箱图标为。`PictureClip` 是一个运行时不可见控件，所以它本身不能显示图片，却可以为其他控件管理和提供图片。该控件将一个大图片裁剪成 m 行 n 列的 $m \times n$ 个大小相同的小图片 (其中 m 和 n 都为大于 0 的整数)。

一个功能较强的程序会提供多个工具栏，每个工具栏上有多个按钮，为这些按钮准备图片是件麻烦的工作，而编程使用这么多的图片也是程序员的“噩梦”。比较好的解决办法是将所有用到的大小相同的图片画在同一幅大图片中 (如图 3.1 和图 3.2 所示)，根据需要可以是一行或一列，也可以是多行多列。有了这样的大图片，使用 `PictureClip` 控件将其裁剪为小图片并赋值给其他控件显示是很容易的。



图 3.1 一个“常用”工具栏用到所有图片组成的大图片



图 3.2 一个“绘图”工具栏用到所有图片组成的大图片

将大图片赋给 `PictureClip` 控件的 `Picture` 属性，然后设置 `Rows` (行数) 和 `Cols` (列数) 属性。通过 `GraphicCell` 属性即可得到指定的小图片。例如：

```
PictureClip1.Picture = LoadPicture("f:\toolbar2.bmp") '指定大图片
PictureClip1.Rows = 10 '10 行
PictureClip1.Cols = 10 '10 列
Picture1.Picture = PictureClip1.GraphicCell(21) '得到小图片
```

这里要注意, `GraphicCell` 属性是一个一维数组, 其下标的有效范围是 $0 \sim \text{Rows} \times \text{Cols} - 1$ 。一个在第 i 行、第 j 列的小图片的下标为 $(i-1) \times \text{Cols} + (j-1)$, 也就是从 0 开始一行一行地计数时的序号。

默认情况下, `PictureClip` 控件将整个图片裁剪。如果用 `ClipX` 和 `ClipY` 属性指定裁剪区域的左上角, 用 `ClipHeight` 和 `ClipWidth` 属性指定裁剪区域的范围, 则可以只裁剪图片的一部分。

如果在设计时通过属性窗口为 `PictureClip` 指定一个大图片, 则这个图片将会被编译到可执行文件中, 程序运行时不再需要原来的图片。

(3) 使用资源文件。资源文件是一种扩展名为 `.res` 的文件, 这种文件可以同时保存多个位图、图标和光标, 甚至还可以保存字符串和其他数据。有专门的程序编辑并生成这种文件 (本书配套光盘中 `Gifts` 文件夹中附带了 Borland 公司的 `Image Editor` 软件, 使用此软件不仅可以编辑、查看资源文件, 还可以编辑位图、图标和鼠标光标文件, 如图 3.3 所示)。资源文件中的每个图片用一个整数或一个字符串来标志。

必须将资源文件添加到工程中之后才能从中读出图片。使用 Visual Basic “工程” 菜单中的 “添加文件…” 命令 (从打开对话框中选择 “资源文件” 类型) 可以将资源文件添加到当前工程中。生成可执行文件时, 资源文件会被包含到可执行文件中。所以, 程序运行时不再需要原来的 `.res` 文件了。使用资源文件还有一个好处在于, 它可以保存大小不同的图片, 这正是 `PictureClip` 控件不具备的。

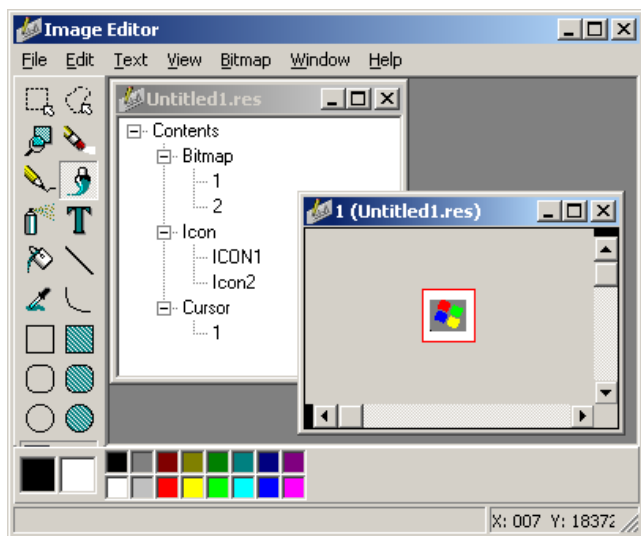


图 3.3 使用 Borland Image Editor 编辑资源文件

一般情况下, 只要知道标志图片的整数或字符串, 便可以使用 `LoadResPicture` 函数将其从资源中读出, 而不必了解其到底是如何存储的。

```
LoadResPicture(index, format)
```

`LoadResPicture` 函数的 `index` 参数是要加载的图片的标志, 可以是整数或字符串; `format` 参数指定打开的资源类型 (见表 3.3)。

表 3.3 format 参数的意义

常数	参数值	意义
vbResBitmap	0	位图资源
vbResIcon	1	图标资源
vbResCursor	2	光标资源

例如，下面的程序定义了一个图片类型的对象数组，然后从资源中加载了一个位图赋值其中一个元素。

```
Dim picBoard(1 To 100) As StdPicuter
Set picBoard(i) = LoadResPicture(100 + i, 0)
```

(4) 使用 ImageList 图像列表控件。ImageList 控件是一个属于 Windows 公共控件的 ActiveX 控件，工具箱图标为，运行时不可见（关于如何添加 ActiveX 控件，请参阅 1.1.9 小节）。该控件主要的作用是为其他的 Windows 公共控件（如 ToolBar 工具栏控件）管理和提供图片，当然也可以用来为一般的控件提供图片。

① 设计时为 ImageList 控件提供图片。通过该控件的“属性页”对话框的“图像”选项卡（见图 3.4 所示）可以插入或删除图片（不支持光标文件）。控件自动按顺序为图片赋一个“索引”值（即 Index 属性），第一个图片索引为 1，以此类推。程序可以通过该索引值访问图片。因为删除和插入图片时，其后所有图片的索引值会重排，所以在程序中使用索引值并不方便。每个图片还有一个字符串类型的属性 Key（即“关键字”，该属性默认为空），应将其设为不同的值来标志每个图片，如使用“Print”来表示“打印”图片。由于字符串的大小写是敏感的，所以在程序中也应使用相同大小写的字符串，否则会出错。

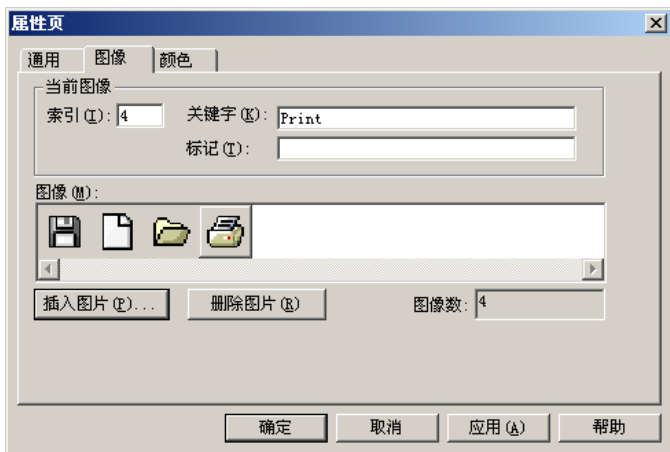


图 3.4 ImageList 控件的“属性页”对话框

通过其他 Windows 公共控件的 ImageList 属性，可将 ImageList 控件与该控件建立联系后，Visual Basic 便不允许对 ImageList 控件中的图片进行修改了，所以应事先将所有可能用到的图片都插入到 ImageList 控件中。

通过属性页插入的图片会随二进制窗体文件 .frx 一起保存且被包含到可执行文件中，程序运行不依赖于原始图片。

② **ImageList** 控件管理的每个图片在运行时会生成一个 **ListImage** 对象，控件的 **ListImages** 属性是所有 **ListImage** 对象形成的集合。通过 **ListImages** 属性可以访问控件管理的每个图片，也可以在运行时动态地添加和删除图片。

ListImage 对象的 **Picture** 属性即为其对应的图片、**Index** 属性为图片的索引（即下标）、**Key** 属性即其“关键字”、**Tag** 属性即其“标记”。

通过索引或关键字可以从 **ListImages** 集合中访问某个 **ListImage** 对象。例如：

```
Form1.Picture = ImageList1.ListImages(4).Picture
Picture1.PaintPicture ImageList1.ListImages("Print").Picture, 100, 0
```

上面第一条语句使用下标（索引）访问 **ListImages** 集合中的一个 **ListImage** 对象，并将其图片赋给窗体的 **Picture** 属性。第二条语句使用关键字“Print”访问 **ListImage** 对象并绘制到图片框中。

也可以使用以下方法获得集合中一个指定的 **ListImage** 对象：

```
Dim imgX As ListImage           '定义 ListImage 类型的对象型变量
Set imgX = ImageList1.ListImages.Item(3) '使用集合的 Item 方法获得指定对象
Image1.Picture = imgX.Picture      '将图片赋值给 Image1 图像控件
```

③ 运行时为 **ImageList** 控件添加图片。使用设计时插入图片的方法只能插入一个一个的、现成的、单独的图片文件，如果想要使用 **PictureClip** 控件从一个大图片中裁剪小图片并把小图片插入 **ImageList** 控件，就只能使用 **ListImages** 集合的 **Add** 方法在运行时添加了。

```
ImageList1.ListImages.Add([index], [key], picture)
```

其中，**index** 参数是新对象的索引，如果省略 **index** 参数则图像添加到最后；**key** 参数指定新对象的关键字；**picture** 参数为对象指定相应的图片。

```
Set imgX = ImageList1.ListImages.Add(1,,LoadPicture("c:\pic.bmp"))
Set imgX = ImageList1.ListImages.Add(,,Picture1.Picture)
```

除了索引之外，**ListImage** 对象所有的属性都可以改变，如：

```
imgX.Key = "Printer"           '更改关键字属性
imgX.Picture = LoadPicture()   '删除对象所对应的图片
```

如果不使用返回值，在调用 **Add** 方法时不能加括号，如：

```
ImageList1.ListImages.Add 2, "Printer",Picture1.Picture
```

④ 运行时删除 **ImageList** 控件中的图片。可以使用 **ListImages** 集合的 **Remove** 方法删除集合中的一个 **ListImage** 对象。

```
object.ListImage.Remove index
```

其中 **index** 是对象的索引或关键字。如：

```
ImageList1.ListImage.Remove "Printer"
```

3.1.4 图片的掩码

除了图片框和图像控件之外，命令按钮、单选框和复选框等控件也可以显示图片，将它们的 Style 属性设置为 1（Graphical），通过 Picture、DownPicture 和 DisablePicture 属性可以为它们指定在一般状态时、被按下状态时和无效状态时所显示的图片。使用这些属性可以制作既有图片显示又有文字说明的漂亮按钮。

但是有一个问题需要注意，一般情况下，一个图片总是有前景内容（即要表现的内容）和背景内容之分的。如图 3.5（a）所示，图片的前景为一个指向右面的三角形，而背景是一个白色的方块。将这个图片放置在灰色的按钮上不太美观，这时需要使用图片的掩码颜色。将按钮的 MaskColor 属性（掩码颜色）设置为图片的背景色（这里为白色），再将其 UseMaskColor 属性（使用掩码颜色）设为 True，如图 3.5（b）所示，图片会忽略与掩码颜色相同的背景色，与按钮浑然一体。



图 3.5 图片掩码颜色的使用

如果使用的是 ico 图标文件，则不存在这种问题，因为图标文件中自动将背景色设置为透明。

3.1.5 鼠标指针样式的设置

鼠标指针形状能表示出程序当前的运行状态，并能给用户更多的信息来掌握一个程序的使用方法，所以在不同时刻为界面上的不同控件或区域设置不同的鼠标指针样式是相当重要的。

在 Visual Basic 中大多数可见对象都具有 MousePointer 和 MouseIcon 属性。MousePointer 属性指定在运行时当鼠标移动到对象上时鼠标指针的形状。此属性的取值见表 3.4。

表 3.4 MousePointer 属性的取值

属性值	常量	鼠标指针的形状
0	vbDefault	（默认值）形状由对象决定
1	VbArrow	箭头
2	vbCrosshair	十字线
3	VbIbeam	I 型
4	vbIconPointer	箭头图标
5	vbSizePointer	四个方向的箭头

续表

属性值	常量	鼠标指针的形状
6	vbSizeNESW	指向右上和左下方向的双箭头
7	vbSizeNS	指向上下方向的双箭头
8	vbSizeNWSE	指向左上、右下方向的双箭头
9	vbSizeWE	左右方向的双箭头
10	vbUpArrow	向上的箭头
11	vbHourglass	沙漏形状（表示等待状态）
12	vbNoDrop	“不能停止”图标
13	vbArrowHourglass	箭头和沙漏
14	vbArrowQuestion	箭头和问号
15	vbSizeAll	四向箭头（表示改变大小）
99	vbCustom	通过 MouseIcon 属性所指定的自定义图标

通过 MousePointer 属性设置的是一些内置的标准鼠标样式，如果无合适的，可以将其设置为 99，然后使用 MousrIcon 属性选择一个以.ico 和.cur 为扩展名的图标文件。例如：

```
Me.MousePointer = 99
Me.MouseIcon = LoadPicture("c:\aaa.ico")           ' 改变窗体鼠标指针
Command1.MousePointer = 13                           ' 改变按钮鼠标指针
```

3.1.6 使用 MMControl 控件播放声音

使用 1.1.9 小节介绍的方法将 ActiveX 控件“Microsoft Multimedia Control”添加到当前工程中。工具箱中增加了图标为的 MMControl 控件。使用该控件可以方便地播放各种音乐和动画文件。

在窗体上添加一个 MMControl 控件，如图 3.6 所示，它由 9 个播放控制按钮组成。按钮的个数及各自的有效性可以通过“属性页”对话框进行设置，如果只是为了播放背景音乐或是提示音，没必要进一步设置，甚至可以将控件的 Visible 属性设置为 False 将其隐藏使之工作于后台。



图 3.6 MMControl 控件

(1) 通过 MMControl 控件的 DeviceType 属性（字符串类型）为其指定要播放的媒体类型。表 3.5 中列出了 MMControl 控件支持的所有多媒体类型，其中较常用的是"AVIVideo"、"WaveAudio"、"Sequencer"和"CDAudio"等。

表 3.5 MMControl 控件 DeviceType 属性的取值

属性值	媒体设备	说明
"AVIVideo"	Microsoft 视频	AVI 格式视频（.avi 文件）
"CDAudio"	音频 CD	用所连接的 CD-ROM 驱动器播放音乐 CD
"DAT"	数字磁带	使用所连接的数字磁带（DAT）设备
"DigitalVideo"	数字视频	数字视频数据
"MMMMovie"	视频	多媒体电影格式（在窗口中显示）

续表

属性值	媒体设备	说明
"Overlay"	视频	帧覆盖设备（在窗口中显示）
"Scanner"	扫描仪	使用所连接的图像扫描仪
"Sequencer"	MIDI 音序器	MIDI 音序器数据
"VCR"	视频磁带	使用所连接的磁带单放机/磁带录像机（在窗口中显示）
"Videodisc"	视盘	使用所连接的视盘机
"WaveAudio"	Microsoft 音频文件	Windows 音频文件（.wav 文件）
"Other"	用户自定义	用户自定义的多媒体类型

（2）通过 FileName 属性指定要播放的文件名及其路径，文件的类型应与 DeviceType 属性值相匹配。

（3）为 Command 属性赋一个字符串类型的指令（见表 3.6）指定 MMControl 控件执行的动作。

表 3.6 MMControl 控件 Command 属性的取值与意义

属性值	意义	属性值	意义
"Open"	打开多媒体设备与文件	"Prev"	前一首
"Close"	关闭多媒体设备与文件	"Next"	下一首
"Play"	播放	"Seek"	查找某一位置
"Pause"	暂停	"Record"	录制
"Stop"	停止	"Eject"	弹出
"Back"	回倒	"Sound"	播放声音
"Step"	步进	"Save"	保存打开的多媒体文件

例如，下面程序段是播放声音文件的一般过程。

```
MMControl1.DeviceType = "WaveAudio"           '指定媒体类型
MMControl1.FileName = App.Path & "\Tada.wav"    '指定文件
MMControl1.Command = "Open"                    '打开媒体
MMControl1.Command = "Play"                    '播放媒体
```

DeviceType 和 FileName 属性可以在设计时设置。

（4）使用 Mode 属性获得 MMControl 控件的当前状态，见表 3.7。

表 3.7 MMControl 控件 Mode 属性的值

属性值	意义	属性值	意义
524	设备未打开	528	设备正在查找
525	设备处于停止状态	529	设备暂停
526	设备正在播放	530	设备就绪
527	设备正在录制		

（5）使用 MMControl 控件的 ButtonClick 事件处理按钮事件。这里的 Button 可以是 Back、Eject、Next、Pause、Play、Prev、Record、Step 和 Stop 中的一个。当用户单击了控

件上的某个按钮，相应的事件就会被触发。此事件过程语法为：

```
Private Sub MMControl1_ButtonClick (Cancel As Integer)
```

这个事件过程有一个 **Cancel** 参数。如果不编写按钮的 **ButtonClick** 事件过程，当用户单击了一个有效的按钮之后，会执行一个预定的操作，如播放、暂停等，这个操作称为默认操作。如果在事件过程中给 **Cancel** 参数赋一个非 0 值，就会阻止对默认按钮操作的执行，只执行事件过程中的语句。如果在此事件过程中不给 **Cancel** 参数赋值，当执行完事件过程之后，还会执行默认操作。

如果将 **MMControl** 控件设为不可见，则不会有上述事件发生，程序不必处理。

(6) 当控件播放结束时会引发 **Done** 事件，事件过程的语法为：

```
Private Sub MMControl1_Done (NotifyCode As Integer)
```

NotifyCode 参数传递的是操作完成的情况。1：播放完毕；2：播放被其他操作中断；4：播放被其他用户终止；8：播放失败。下面的程序段在音乐播放完成时关闭媒体文件。

```
Private Sub MMControl1_Done (NotifyCode As Integer)
    MMControl1.Command = "Close"
End Sub
```

3.1.7 使用 ProgressBar 进度条控件

ProgressBar 进度条控件是一个 Windows 公共控件（ActiveX 控件），使用 1.1.9 小节介绍的方法将“Microsoft Windows Common Controls”添加到工程中，其中图标即为进度条控件。

进度条控件一般用来指示需要较长时间完成任务的进度，如软件安装、耗时的计算、数据的传输等。与其他的公共控件相比，进度条的用法比较简单。进度条控件主要的属性有 **Max**（最大值）属性和 **Min**（最小值）属性以及 **Value** 属性，与滚动条控件的同名属性意义相同。前两个属性可以在设计时设置，**Value** 属性只能在运行时设置并且必须在 **Max** 和 **Min** 属性值之间。在运行时不断改变其 **Value** 属性的值可以实现进度条指示值的变化。

Orientation 属性（方向）为 0 时进度条水平放置，1 时垂直放置。**Scrolling** 属性为 0 时进度条是分块的（见图 3.7 下），1 时为连续的（见图 3.7 上）。

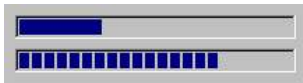


图 3.7 进度条控件

进度条控件可以响应 **Click**、**MouseMove**、**MouseUp** 和 **MouseDown** 等事件。因为进度条应该是个“纯输出”控件，所以一般不必编写其事件过程。

3.2 题目 15 图片浏览器☆

难度：☆。知识点：（1）文件系统控件；（2）定时器控件；（3）图片的应用；（4）多模块程序设计。

1. 题目介绍

多数人都有大量的计算机图片文件，有扫描的、数码相机拍摄的、网上下载的，有生活照、风景照和科技照。人们习惯将这些照片分类保存在不同的文件夹中。本题目要求编制一个“图片浏览器”程序，能够浏览保存在文件夹中的图片文件。

2. 功能要求

（1）如图 3.8 所示，程序启动之后，单击“指定文件夹”按钮，弹出“指定文件夹”对话框，从中选择计算机中某个驱动器上的一个文件夹。单击“确定”之后，程序自动搜索指定文件夹中扩展名为.jpg、.bmp、.ico、.cur 和.gif 的图片文件，将图片文件名显示在主窗口左边的列表框中，并将当前的图片文件夹名显示在主窗口底部（如图 3.9 所示）。

（2）指定了图片文件夹之后，程序在主窗口右部的显示区显示第一幅图片。这时，使用鼠标单击列表中的文件名、使用键盘上的方向箭头键、使用工具栏上的“前一幅图片”和“下一幅图片”按钮都可以显示不同的图片。使用“第一幅图片”和“最后一幅图片”按钮可以显示列表中的第一幅和最后一幅图片。

浏览图片时，窗口左下角显示图片总数以及当前显示图片的序号。

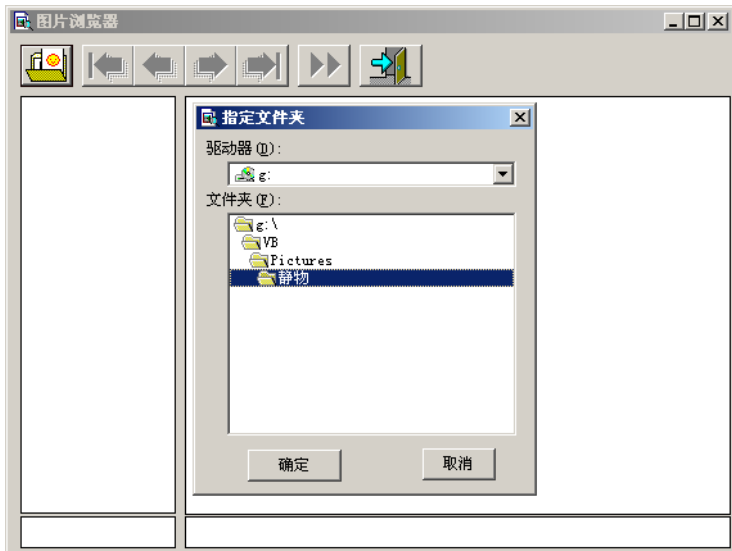


图 3.8 指定保存有图片的文件夹

(3) 单击工具栏上的“自动播放”按钮，程序自动以每秒一幅的速度以幻灯片形式循环显示所有图片。在自动播放时，鼠标、键盘和工具栏上其他按钮的功能仍可以正常使用。自动播放时，“自动播放”按钮变为，单击停止自动播放。单击“退出”按钮关闭本程序。

(4) 为了最大限度地使用屏幕的显示能力，要求程序主窗口可以通过拖动边框、单击“最大化”、“最小化”和“还原”按钮来改变其大小。主窗口改变大小时，上面所有的控件都应调整其大小与位置来保证正常的显示效果。

(5) 工具栏上的按钮根据程序的运行状态自动变为有效或无效。例如，在程序未指定文件夹时，中间五个按钮全为灰色无效状态（如图 3.8 所示）；如果显示的是最后一张图片，则“下一幅图片”和“最后一幅图片”按钮应为无效。

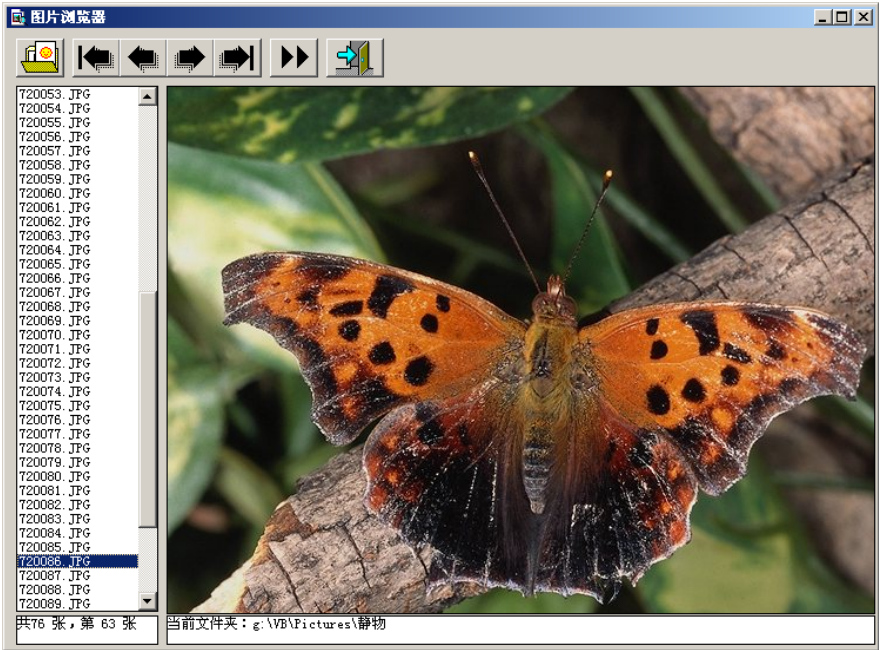


图 3.9 浏览图片

3. 难点与提示

(1) “指定文件夹”对话框中使用了 Visual Basic 的内部控件“驱动器列表框”和“目录列表框”，能够显示计算机中所有的驱动器和文件夹。

(2) 显示图片可以使用图像（Image）或图片框（PictureBox）控件。因为图片框控件所占用的系统资源比图像控件大，故不宜使用图片框。

(3) 搜索某个文件夹中的图片文件，可以使用 Dir 函数（参见 2.1.4 小节）。

4. 更上一层楼

如果一个图片非常大，即使将窗口最大化也不能完全显示而又不希望将其缩小，样例程序就无能为力了。试着增强程序，可以使用滚动条平移大图片进行观察。

3.3 题目 16 快速配对游戏☆☆

难度：☆☆。知识点：（1）图像控件；（2）定时器控件；（3）顺序文件的读写；（4）控件数组；（5）图片的使用。

1. 题目介绍

“快速配对”是一个简单又有挑战性的小游戏。窗口中有 16 张图片，共 8 种图案，一种图案有两张。游戏开始时程序向用户展示所有的图片，片刻后图片翻过去。用户凭记忆连续单击两张自己认为图案相同的图片将其翻起，如果被单击的图片不相同，则会自动翻回去。这样不断单击，直到所有图片均被翻过来，使用时间越少越好。

2. 功能要求

（1）运行程序，显示如图 3.10 所示的开始界面，图片均“背对”用户。单击“开始”按钮，图片显示 1.5s（如图 3.11 所示，图片随机排列），然后翻过去。用户使用鼠标单击图片将其翻起，如果连续单击的两个图片内容相同，则会保持显示状态；如果两图片不同，两张图片 0.5s 之后自动翻过去。用户凭记忆在尽量短的时间内将 8 对图片全部“翻起”，游戏完成。

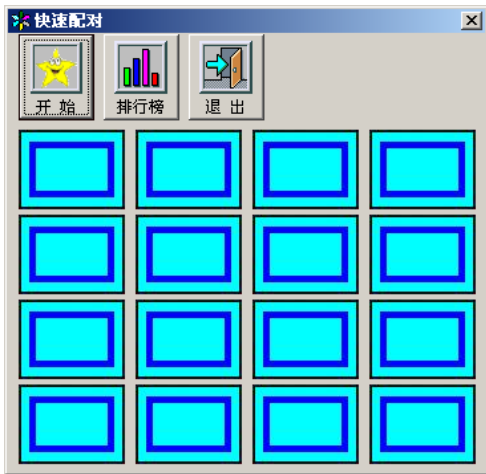


图 3.10 程序开始状态



图 3.11 程序展示所有图片

（2）从用户第一次单击图片时，程序开始计时（以秒为单位），并在窗口的右上角显示已用时间（见图 3.12）。每击中一对图片，窗口中显示一个“笑脸”图标（图 3.12），否则显示一个“严肃的脸”图标。

（3）程序自动记录前三个最短完成时间，在游戏结束时以图 3.13 所示的消息框显示名次。单击窗口中的“排行榜”按钮可以弹出“排行榜”对话框，如图 3.14 所示，显示前三名所用的时间。

（4）在游戏过程中，窗口的“开始”和“排行榜”按钮呈无效状态，游戏完成后，“开

始”按钮变为“重来”，单击该按钮可以从头再来一局。

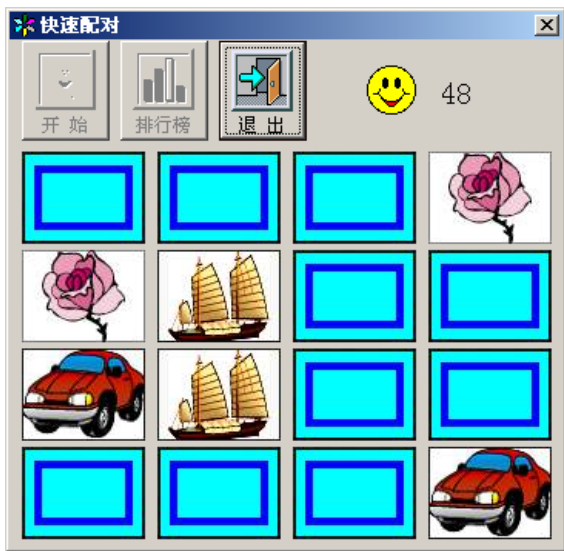


图 3.12 快速配对游戏



图 3.13 显示名次消息框图

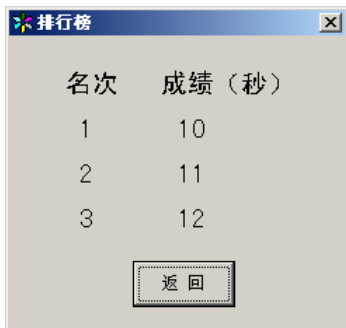


图 3.14 “排行榜”对话框

3. 难点与提示

(1) 图 3.15 显示了本程序设计时的窗体及控件，其中 Image1 控件数组用来显示 16 张图片，Image2 控件数组保存要用到的 8 种内容的图片，ImageBack 保存背面图片。因为在游戏过程中图片会不断变化，如果临时从磁盘读入既影响速度又会造成系统忙碌，所以可以在设计时就把用到的图片赋值给 Image2 控件数组和 ImageBack 控件，在程序需要时将其赋值给 Image1 数组中的每个控件。Resource 文件夹中的 pic01.jpg~pic31.jpg 可以用作本程序的图片。

同理，“笑脸”和“严肃的脸”两个图片（分别为 face01.ico 和 face02.ico 文件）也可以使用图像控件保存，在游戏过程中根据情况显示相应的图片。

Image2 控件数组、ImageBack、ImageSmile、ImageNosmile 控件在程序过程中是隐藏的。

(2) 本题目有三个地方需要用到 Timer 定时器控件，一是 1.5s 的图片展示、二是两个

图片不相同时的 0.5s 延迟,三是游戏时间的记录。为了便于编程,可以使用三个独立的 Timer 控件。

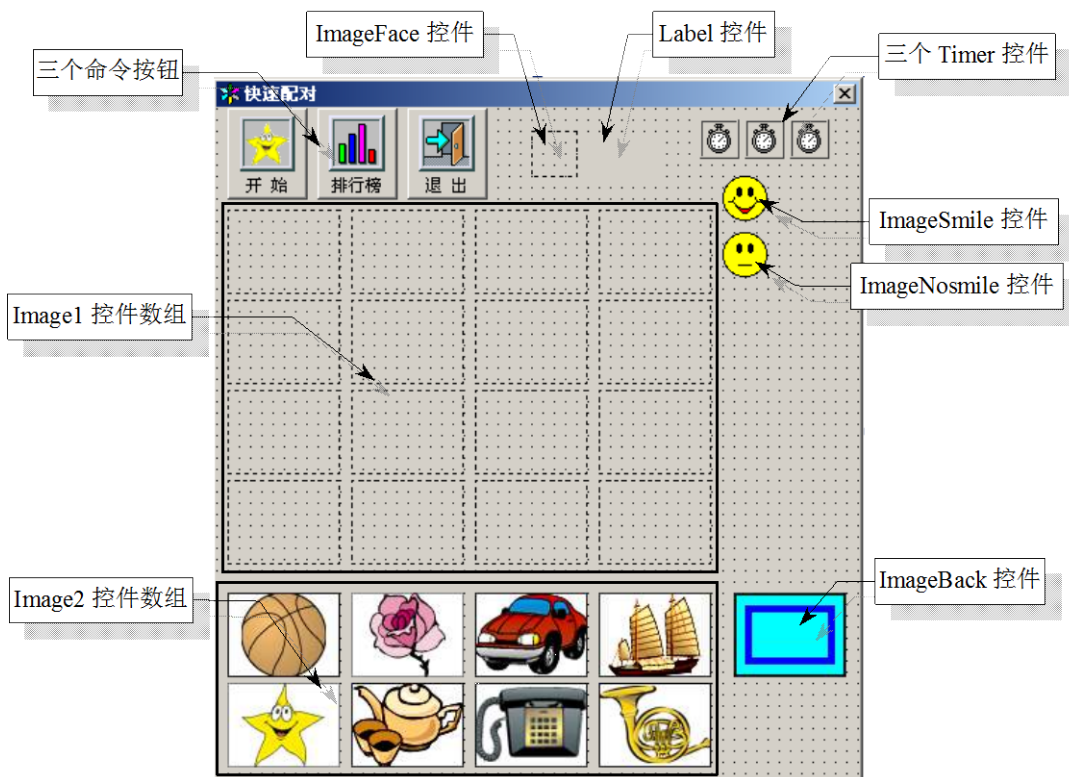


图 3.15 设计时的窗体

(3) 本题目的难点是如何随机地显示 16 个图片, 这里给出一种方法。定义如下 Arrange 数组来保存每个位置上的图片。

```
Private Arrange(0 To 15) As Integer '保存每个位置的图片
```

使用以下程序将两组 0~7 共 16 个整数随机地赋值给上述数组的 16 个元素。

```
Randomize
For i = 0 To 15
    Arrange(i) = -1
Next
For i = 0 To 7
    For j = 0 To 1
        Do
            m = Int(Rnd * 16)
            If Arrange(m) = -1 Then
                Arrange(m) = i
                Exit Do
            End If
        Loop
    Next j
Next i
```

```

        End If
    Loop
Next
Next

```

Arrange 数组在 Image1 和 Image2 两个控件数组之间建立了联系。例如,若 Arrange(10)=5 表示 Image1 中下标为 10 的控件显示 Image2 中下标为 5 控件上的图片。下列程序段可以将 16 张图片全部显示出来。

```

For i = 0 To 15
    Image1(i).Picture = Image2(Arrange(i)).Picture
Next

```

(4) 为了减少出错的可能性,那些已经被翻开的图片应不能再被单击,或者说不再响应单击的操作。比较好的办法是定义一个如下的逻辑型数组 Finished,被翻开的图片所对应的元素赋值为 True。当响应鼠标事件时,可根据该数组的值来判断被单击的图片是否已被翻,并分别对待。

```

Private Finished(0 To 15) As Boolean

```

(5) 本程序将前三名所用时间保存在名为 Record.txt 的文本文件中,可以在“排行榜”中显示,也可以在新的记录出现时更新。

4. 更上一层楼

(1) 本题目样例程序所显示的 8 种图片内容是固定不变的,玩久了会令人生厌,试着将程序扩充,使得用户可以指定图片(比如从提供的 31 张图片中任选 8 张)。

(2) 本例的“排行榜”功能只记录和显示前三名的时间,请增强程序使之可以记录用户的姓名。

3.4 题目 17 多功能计时器☆☆

难度:☆☆。知识点:(1)日期时间函数;(2)格式转换函数 Format;(3)PictureClip 控件;(4)图片绘制;(5)多模块程序设计;(6)菜单的使用。

1. 题目介绍

本题目要求设计一个具有时钟、秒表和倒计时功能的液晶显示式“多功能计时器”。

2. 功能要求

(1) 程序启动时,首先进入如图 3.16 所示的“时钟”模式。要求以“YYYY-MM-DD HH:MM:SSA/P”的形式显示当前日期、时间和上下午。月日時分秒始终以两位数字显示。日期和时间之间留一空格,小时采用 12 小时制,用“A”表示上午、“P”表示下

午。图 3.16 所示的时间“2003-9-26 09:32:55A”表示上午 9 点 32 分，图 3.17 表示下午 7 点 35 分。

(2) 从“文件”菜单中选择“秒表”或“倒计时”命令可以进入相应的模式。菜单项前的“√”标记用来指示当前模式。

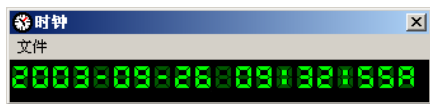


图 3.16 时钟模式

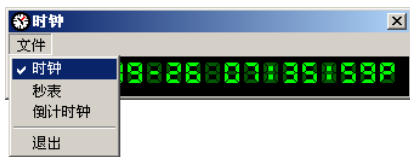


图 3.17 主菜单

(3) 选择“秒表”模式后，程序显示“0:00:00.00”（如图 3.18 (a) 所示），单击“开始”按钮后秒表走时，小数点后精确到 0.01s，从“.00”~“.99”。

秒表走时时，如图 3.18 (b) 所示，“开始”按钮变为“暂停”，“回零”按钮成为无效状态。单击“暂停”按钮，秒表暂停，“暂停”按钮变为“继续”，单击此按钮秒表接着暂停前的时间走时。在暂停状态时，“回零”按钮变为可用（如图 3.18 (c) 所示），单击将秒表示数归零，“继续”按钮变为“开始”。



(a)



(b)



(c)

图 3.18 秒表模式

(4) 选择相应菜单命令进入倒计时模式时，首先弹出如图 3.19 所示的“设置倒计时长度”对话框。输入时分秒值并单击“开始”按钮关闭对话框后直接开始倒计时（如图 3.21 所示）。如果指定的时间不正确，如输入负值或大于 59 的分或秒，则显示图 3.20 所示的消息框要求重新输入。

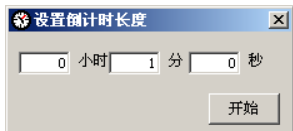


图 3.19 设置倒计时时间长度



图 3.20 错误信息

如果倒计时时间归零，则显示图 3.22 所示的“时间到!”字样，并且窗口背景以不同的颜色闪烁以示提醒。



图 3.21 倒计时



图 3.22 倒计时时间到

3. 难点与提示

(1) 本题目的 Resource 文件夹中提供了一个图片文件 LED.BMP (如图 3.23 所示), 该图片的大小为 256×27 像素。图片中依次共有 0~9、冒号、连字符、点、A、P 和空格等 16 个本程序需要用的 LED 风格字符, 每个字符宽度均为 16 个像素。



图 3.23 数码管方式使用的图片

使用 PictureClip 控件将此图片分为 16 个图块 (Rows 属性为 1, Cols 属性为 16), 然后根据显示的需要将图块绘制到窗体或图片框中即可显示时间, 也可以将图块赋值给 Image 控件数组来显示时间。

下面的例程演示了如何将一个包括时间信息的字符串以液晶的效果显示到一个图片框中, 其中 PC 是已设置好 Picture、Rows 和 Cols 属性的 PictureClip 控件名。

```
Private Sub Show_LED(pic As PictureBox, str As String)
    Dim s As String * 1
    Dim pos As Integer
    Dim i As Integer
    Dim n As Integer

    For i = 1 To Len(str)
        s = Mid(str, i, 1)
        n = -1
        Select Case s
            Case "0" To "9"
                n = CInt(s)
            Case ":"
                n = 10
            Case "-"
                n = 11
            Case "."
                n = 12
        End Select
        pic.PictureClip = Image1(n)
    Next i
End Sub
```

```

        Case "a", "A"
            n = 13
        Case "p", "P"
            n = 14
        Case " "
            '这是空格
            n = 15
    End Select
    If n <> -1 Then
        pic.PaintPicture PC.GraphicCell(n), pos, 0
        pos = pos + 16
    End If
Next
End Sub

```

因为所有显示内容全部来自于 LED.BMP 文件，所以可以在设计时将此文件赋值给 PictureClip 控件的 Picture 属性。关于 PictureClip 控件的用法，参见 3.1.3 小节。

(2) 为了保证正确绘制，应将显示时间的图片框（或窗体）的 ScaleMode 属性设置为 3（以像素为绘图单位）、将 AutoRedraw 属性设计为 True。

4. 更上一层楼

样例程序在每次 Timer 事件发生时全部重绘所有的字符，这在编程上相对简单，但是程序的执行效率不高。例如，时钟由“2003-09-26 09:32:55A”一秒后变为“2003-09-26 09:32:56A”，其实只有秒发生了变化，没有必要将所有字符重绘。这样会大大浪费计算资源，在秒表模式下问题会更突出，因为每隔 0.01s（即 10ms）刷新一次，如果计算机速度较低，可能导致此功能无法实现。因此在编制绘制字符的程序时，应尽量只重绘那些变化了的字符。

3.5 题目 18 存包柜☆☆

难度：☆☆。知识点：（1）控件数组；（2）随机文件读写；（3）PictureClip 控件；（4）图片应用。

1. 题目介绍

在有些公共场所（如超市）的入口处常有一些自动存包柜。客户按动存包按钮，机器自动打开一个空着的小柜子的柜门，并打印出一张印有密码的纸条。客户把包（或其他物品）放到小柜子中，关上柜门即完成存包操作。

当客户要取包时，用存包柜上的数字按钮输入存包时打印的密码，相应小柜门自动打开，客户可以取走自己存放的物品。

本题要求编写一个程序来模拟这种自动存包柜的功能。

2. 功能要求

(1)存包柜界面如图 3.24 所示,存包柜上有 24 个小门,即 24 个小柜子,编号为 01~24。24 个小指示灯指示当前柜子占用情况,亮灯表示对应的小柜子已有物品存放。

中间 LED 数字可显示柜子和输入的密码,下面的数字按钮用于用户输入密码。

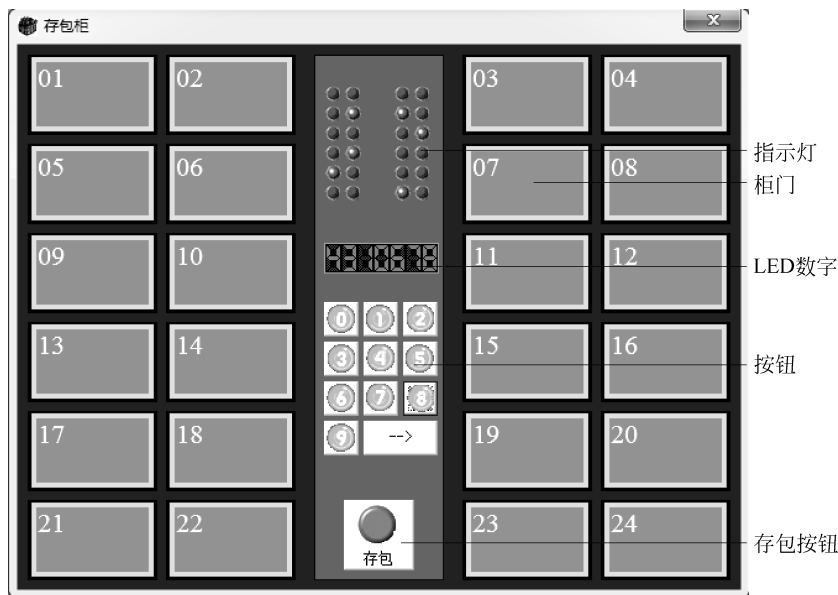


图 3.24 存包柜界面

(2) 当要存放物品时,用户按动中间底部的大“存包”按钮,程序显示如图 3.25 所示的消息框,告诉用户箱子号(柜号)和密码,并自动打开相应的小柜门,用户单击柜门上的“放入物品”按钮,程序弹出“选择物品”对话框,如图 3.26 所示。

用户单击一样物品的图片,表示存入此类物品,然后单击如图 3.27 所示的“关闭柜门”按钮,存包操作完毕。

存包后,用户可以忘记自己的物品存放到哪个小柜子中,但必须记住自己的密码。

(3) 客户想要取出物品时,直接通过按钮输入密码,密码正确,柜门自动打开,LED 数字区显示柜子编号,如图 3.28 所示。

物品取出后,柜子空了,相应的指示灯灭掉,此柜子可供其他人使用。

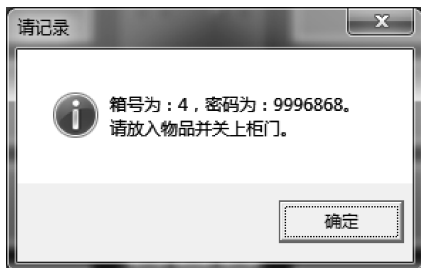


图 3.25 显示密码



图 3.26 “选择物品”对话框

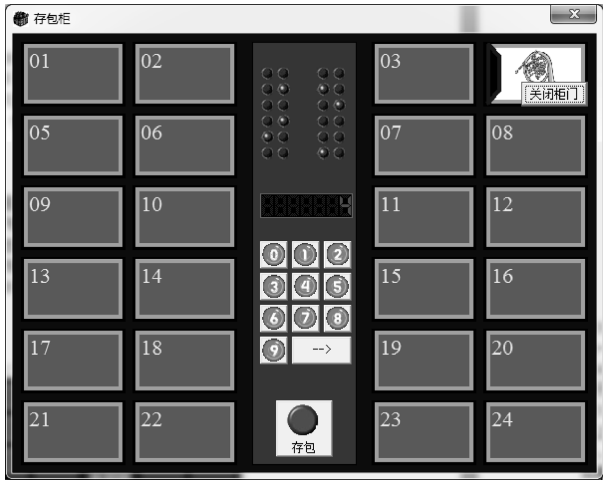


图 3.27 “关闭柜门”按钮

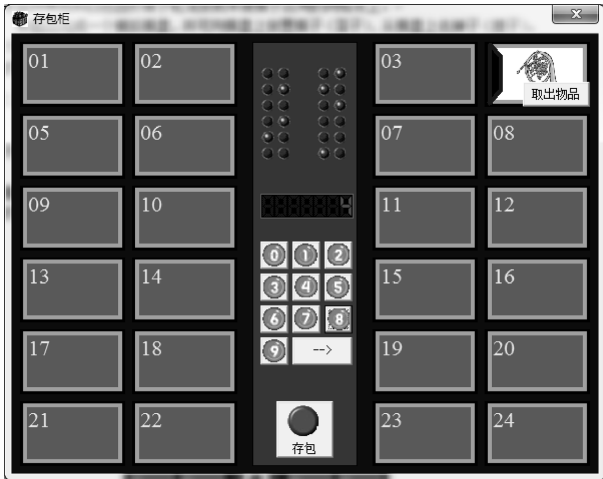


图 3.28 “取出物品”按钮

3. 难点与提示

(1) 如图 3.29 所示, 24 个小柜门可以使用图像控件数组实现, 在不同的时刻加载表示关闭和开启的两个图片即可。柜门上面的编号可使用标签控件数组实现。

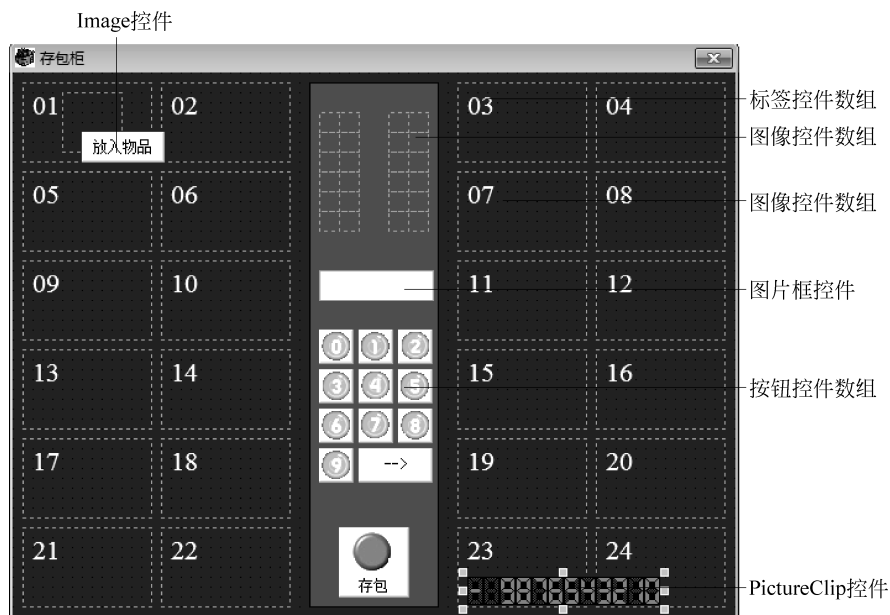


图 3.29 设计状态界面

24 个指示灯用图像控件数组实现。10 个数字按钮可使用按钮控件数组实现, 需要将其 Style 属性设置为 1, 单独设置其 Picture 属性指定数字图片。

LED 形式显示数字的功能由一个图片框控件的绘图功能实现, 单个数字字符来自于 PictureClip 控件, 该控件管理了 0~9 共 10 个数码管形式的图片。

上述的这些图片程序已提供。

(2) 为了记录每个柜子是否被占用, 以及占用柜子中放的物品类型和密码, 可以定义如下的自定义数据类型。

```
Public Type Info
    Occupied As Boolean      '是否被占用
    Code As String * 7      '7 位密码
    Goods As Integer        '物品编号
End Type
```

再定义 Info 类型的数组用来存储所有柜子的当前状态。

```
Public Infos(0 To 23) As Info
```

当程序关闭时(窗体的 Unload 事件过程), 将 Infos 数组的值以随机文件的形式保存到 data.dat 文件中。下一次运行程序时, 在 Load 事件过程中读入此文件的内容, 可保证存包

的信息不会因程序的重启而丢失。

(3) 为了使操作简化, 本程序指定了 9 种固定的存放物品, 如图 3.26 所示, 弹出的对话框是另一个窗体模块, 使用了 9 个图像控件数组来显示物品图片。用户存包时只需单击选择即可, 存储时只记录其编号 (0~8) 即可。

(4) 本程序编程时需要注意的是: 客户存包时, 打开的必须是空柜子; 生成的密码必须是唯一的, 输入密码正确时必须打开相应的柜子, 否则是功能性失败。

(5) 本程序共有以下几个状态: 就绪 (既可按存包按钮, 也可以按数字按钮取包)、存包柜门打开状态 (等待放入物品)、选择物品状态、选完物品未关柜门状态、打开柜门状态 (等待取出物品)。程序设计时必须使用变量记录不同的状态, 否则程序的功能会混乱。

4. 更上一层楼

(1) 增强程序, 使客户可以随意指定要存放的物品 (指定磁盘上现存的图片)。

(2) 增强程序, 可统计一段时间内 (如一天) 每个柜子的使用情况。

3.6 题目 19 拼图游戏☆☆☆

难度: ☆☆☆。知识点: (1) 控件数组; (2) 公共对话框控件; (3) PictureClip 控件; (4) 多模块程序设计; (5) 图片的应用。

1. 题目介绍

“拼图”是一种老少皆宜、容易上手的益智类小游戏, 本题目要求编制一个拼图游戏程序。该程序的使用方法如下: (1) 指定一幅完整的图片, 将其分割为 $m \times n$ 个小图块并打乱顺序后重新排列, 其中有一个没有图块的“空当” (如图 3.30 所示)。(2) 使用鼠标单击与空当相邻的图块使该图块与空当位置互换。(3) 利用空当位置移动各图块, 最终恢复图片原貌, 完成拼图。移动的步数越少表明游戏者的水平越高。游戏过程中, 可以打开参考图帮助查找各图块的正确位置。

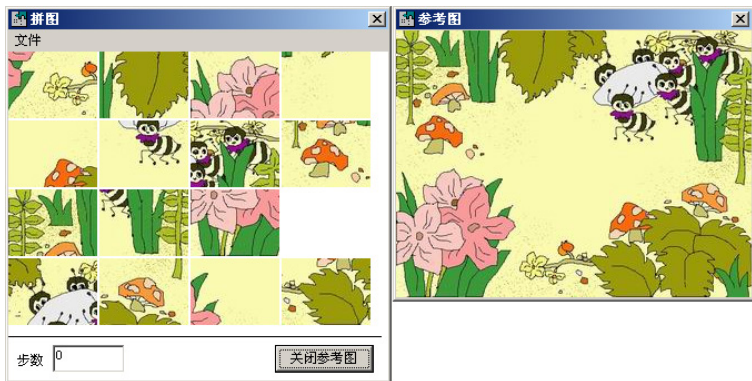


图 3.30 拼图游戏

2. 功能要求

(1) 运行程序, 显示如图 3.31 所示的开始界面。为了使打开图片之前的界面更生动, 在进入游戏前显示一个背景图像。



图 3.31 游戏开始界面

(2) 选择菜单中的“打开图片”命令, 显示如图 3.32 所示的“打开图片”对话框要求指定一个图形文件。“打开图片”对话框应使用公共对话框控件实现, 并且只能指定以*.JPG和*.BMP为扩展名的文件。用户从对话框顶部的“查看”菜单中选择“缩略图”方式, 可以在打开图片之前对图片进行预览。



图 3.32 打开图片文件

(3) 选择图片之后, 程序弹出如图 3.33 所示的“指定行列数”对话框, 提示将图片分几行几列。使用文本框后面的微调按钮可以调节行数和列数, 行列数均限制在 3~7 之间。



图 3.33 指定图块数

(4) 指定行列数之后, 单击对话框的“确定”按钮返回

主窗口，如图 3.34 所示。程序自动按指定的行列数将图片分割成图块，然后随机排列。

(5) 打乱的图片中有一空当位置，用户单击与空当位置相邻的图块可以与其互换位置。若单击了空当位置或不与空当相邻的图块，程序不做任何操作。使用此方法移动所有图块直至恢复图片的原貌。原图片只缺少最右下角的图块（如图 3.35 所示）。



图 3.34 游戏过程中

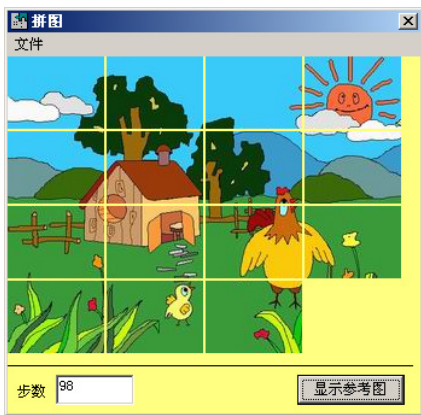


图 3.35 复原后的图片

(6) 图片复原时，程序显示消息框表示祝贺（如图 3.36 所示），同时显示游戏共使用的步数。

(7) 游戏过程中，如果用户希望参考原图，可以单击“显示参考图”按钮，打开“参考图”窗口（见图 3.30）。要求“参考图”窗口与主窗口靠在一起且上边框对齐。“参考图”窗口打开时，按钮的标题显示为“关闭参考图”，再单击此按钮可以关闭“参考图”窗口。也可以通过单击其右上角“关闭”按钮来关闭“参考图”窗口。

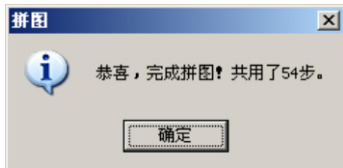


图 3.36 游戏完成时的消息框

(8) 因为空当位置和图块之间的间隙直接显示了窗体背景颜色，如果此颜色与当前图片主体颜色很接近，会干扰拼图操作，所以要求窗体背景色可更改。从菜单中选择“背景颜色”命令，弹出“颜色”公共对话框（见图 2.2），用户从中选择一新颜色即可。

(9) 在游戏过程中或在完成后，均可以退出程序或选择“打开图片”菜单命令重新开始一个新图片。

(10) “参考图”窗口应是一个非模态窗口，在关闭主窗口退出程序时，应确保该窗口同时卸载，而不是隐藏，否则该程序不会完全关闭，仍会驻留内存。

3. 难点与提示

(1) 本题目的 pics 文件夹中提供了 20 个图片文件：pic_01.jpg~pic_20.jpg，可以用作拼图图片，所有图片的大小均为 320×240 像素。Resource 文件夹中的 bkgrnd.jpg 可作开始界面。

(2) 程序的主窗体可以按照图 3.37 所示进行设计。PictureClip 控件用来管理打开的图片并将其分为指定的行列图块，其 Picture、Rows、Cols 等属性值应在程序中根据用户的

选择进行设置,设计时不必指定。关于 PictureClip 控件的使用方法,参阅 3.1.3 小节。

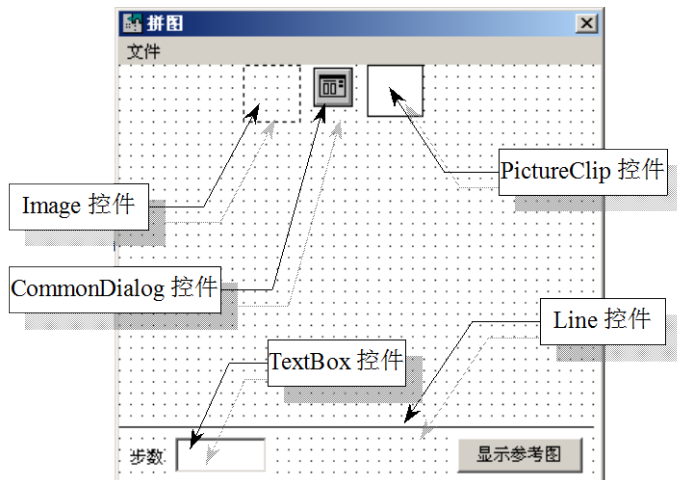


图 3.37 设计时的主窗体

(3) 使用 Image 控件数组来显示图块比较合适。设计时,只需在窗体上放置一个 Image 控件并将其 Index 属性设置为 0,其他的控件数组元素在运行时根据用户指定的行列数使用 Load 语句加载。关于控件数组的用法,参阅 1.1.8 小节。

(4) “打开图片”和“颜色”对话框均由公共对话框 CommonDialog 控件实现。在显示“打开图片”对话框时,应设置其 Filter 属性使之只能打开 BMP 和 JPG 类型的图片文件。CommonDialog 控件的用法,参阅 2.1.6 小节。

(5) 应适当设置主窗口、“参考图”窗口和“指定行列”对话框的边框类型(即 BorderStyle 属性)。在加载显示“参考图”窗口和“指定行列”对话框时应选择适当的模态/非模态参数(窗体的 Show 方法)。

(6) 因为在用户使用本程序时可能打开很大或很小的图片文件,为了使程序正常显示,有必要将 Image 控件的 Stretch 属性设计为 True,使之可以自动将大图块缩小、把小图块放大。这样,主窗口的大小可以设为一个定值,根据行列数来调整 Image 控件的大小即可。

(7) 本题目的算法重点有三个:如何随机排列图片、如何判断被单击的图块是否与空当相邻、如何判断拼图是否完毕。

假设 CommonDialog 控件名为 CD、PictureClip 控件名为 PC、Image 控件数组名为 imgPT,并在主窗体的声明段有如下代码(将 Rows 和 Cols 声明为全局变量是因为行列数要在另一个窗体中指定)。

```
Public Rows As Integer, Cols As Integer      '图块的行数和列数
Dim Arrange() As Integer                    '图像控件与数组的对应关系
Dim space As Integer                        '空当的位置
```

PictureClip 控件的 GraphicCell 属性是一维数组,设置了 Picture、Rows 和 Cols 属性之后,GraphicCell 属性的下标为 $0 \sim \text{Rows} * \text{Cols} - 1$,所以,Arrange 数组可以重定义为 Arrange(0 To Rows*Cols-1)。使用 Arrange 数组可以在 Image 控件数组和 PictureClip 的 GraphicCell

属性数组之间建立桥梁。Arrange 数组的下标代表相应的 Image 控件数组元素，Arrange 数组的值代表相应的 GraphicCell 属性数组元素。例如：

```
Arrange(3)=8
imgPT(3).Picture=PC.GraphicCell(Arrange(3))
```

表示下标为 3 的 Image 控件数组元素显示下标为 8 的图块。所以，如果将 0~Rows*Cols-1 这 Rows*Cols 个整数随机赋值给 Arrange 的 Rows*Cols 个元素便可以将图片“打乱”。下面的例程演示了如何随机排列图块。

```
Private Sub Rnd_Arrange() '随机排列图块
    Dim i As Integer, j As Integer : Dim n As Integer
    n = Rows * Cols
    ReDim Arrange(0 To n - 1)
    For i = 0 To n - 1
        Arrange(i) = -1 '用-1 标记未赋值的元素
    Next
    i = 0
    Do '给数组 Arrange 随机赋值
        j = Int(Rnd * n) '产生 0~n-1 的随机数
        If Arrange(j) = -1 Then
            Arrange(j) = i
            i = i + 1
            If i = n Then Exit Do
        End If
    Loop
    For i = 1 To n - 1 '加载控件数组元素
        Load imgPT(i)
    Next
    For i = 0 To n - 1
        If Arrange(i) <> n - 1 Then '将图块赋给控件数组元素
            imgPT(i).Picture = PC.GraphicCell(Arrange(i))
        Else '将"打乱"前图片右下角的图块设为空
            imgPT(i).Picture = LoadPicture()
            space = i
        End If
    Next
End Sub
```

下面的例程演示了当单击某图块时，如何判断其是否与空当相邻并移动图块。

```
Private Sub imgPT_Click(Index As Integer)
    Dim r1 As Integer, c1 As Integer
    Dim r2 As Integer, c2 As Integer
    Dim n As Integer, I As Integer
    c1 = space Mod Cols '计算空当的行列
```

```

r1 = space \ Cols
c2 = Index Mod Cols           '计算单击的行列
r2 = Index \ Cols
If Abs(c1 - c2) = 1 And Abs(r1 - r2) = 0 Or _
    Abs(c1 - c2) = 0 And Abs(r1 - r2) = 1 Then '判断是否相邻
    n = Arrange(Index)
    Arrange(Index) = Arrange(space)
    Arrange(space) = n
    imgPT(space) = PC.GraphicCell(Arrange(space))
    imgPT(Index) = LoadPicture()
    space = Index
End If
End Sub

```

如果 `Arrange` 数组的每个元素值与其下标相同，则可以断定拼图完毕。下面的函数判断是否拼完。

```

Private Function isOK() As Boolean           '判断是否拼完
    Dim i As Integer
    For i = 0 To Cols * Rows - 1
        If Arrange(i) <> i Then Exit For
    Next
    If i = Cols * Rows Then isOK = True
End Function

```

这种方法的优点是：移动图块时不必移动相应的 `Image` 控件数组，只需交换它们所显示的图块。

4. 更上一层楼

(1) 增加“排行榜”功能。记录每个用户的用户名、难度（即行列数）和成绩（即步数）以供查询。

(2) 程序退出时，记住用户的背景色设置和是否显示参考图，下次启动时自动恢复前一次设置。如果退出时游戏未完成，下一次启动时，可以继续上次游戏。

3.7 题目 20 动物管理员游戏☆☆☆☆

难度：☆☆☆☆。知识点：(1) `PictureClip` 控件的使用；(2) `Timer` 定时器控件的使用；(3) `MMControl` 多媒体控件的使用；(4) `ProgressBar` 进度条控件的使用。

1. 题目介绍

“动物管理员”是一款非常有趣的游戏，有很多的变种。本题目要求编制一个玩法相对

简单，但是编程却并不简单的版本。如图 3.38 所示，程序的界面分为三部分，左边部分是一个由 8 行 8 列动物图标组成的游戏区，使用者通过鼠标单击两个相邻的图标使之交换位置，如果交换位置之后某种同类动物图标在同一行或同一列上出现三个或三个以上相邻的情况，表示这些相邻的动物被“捕获”。最右边是成绩显示区，显示出当前共捕获了每种动物各多少只以及总的数量。游戏区和成绩显示区的交接处是时间显示区，在游戏过程中剩余时间越来越少，深色部分越来越短，直到游戏结束。游戏结束时，捕获的动物越多则水平越高。



图 3.38 游戏过程中

2. 功能要求

(1) 程序启动时，游戏区随机分布各类动物图标。图标共有 7 种，如图 3.39 所示，依次是长颈鹿、狮子、猩猩、大熊猫、河马、鳄鱼和大象。每种动物有两种表情，上面一行是未被捕获时的表情，下面一行是被捕获后的表情。

程序启动时，有可能出现同一行或同一列上有三个以上同种动物相邻，这种情况也应算成绩。



图 3.39 7 种动物图标

(2) 用户使用鼠标单击图标时，被单击的图标变红。交换位置之后，图标的颜色变为正常。

(3) 如果用户单击了不相邻的图标，程序发出“错误音”且不变换位置。如果单击的是相邻图标但交换位置后并不能使三个以上同种图标相邻，则会交换回原来位置同时发出“错误音”。

如果交换位置后出现相邻图标，所有满足相邻关系的图标以被捕捉的表情闪烁后消失，形成的空位由上方的图标下落后填补，最上面形成的空位由随机生成的图标填补。消失的图标个数会作为成绩加到右边的成绩显示区，同时播放悦耳的“成功声”。

(4) 很多情况下，往往会有多个相邻关系同时形成。如图 3.40 (a) 所示，五个熊猫形成两个相邻关系，这种情况下的成绩只能加 5。如图 3.40 (b) 所示，在不同位置同时形成多个相邻关系。所以，程序应即时判断游戏区形成的所有相邻关系，并进行处理和加分。



图 3.40 两个相邻关系同时存在的情况

有时，“消掉”相邻图标，上方的图标下落后，使得原来不相邻的图标形成了相邻关系。这时应继续判断和处理整个游戏区的图标相邻关系，直到所有的图标都不构成三三相邻的关系。

(5) 程序游戏时间共 2 分钟，游戏过程中每秒钟发出一声时钟的“滴哒声”，并且时间显示区的蓝条越来越短。游戏时间到时，发出闹钟的“闹铃声”，游戏区所有图标的背景变为红色，并显示出“Game Over”字样。这时游戏结束，鼠标不能再单击图标，如图 3.41 所示。

(6) 为了清楚地表达图标的交换、图标的消去和下落等动作，程序使用了延时。例如，单击两个相邻图标后，隔 200ms 才交换它们的位置，判断相邻后，隔 200ms 才消去，再隔 200ms 才将上面的图标下移……在延时过程中，用户的鼠标单击是无效的，这时鼠标指针变为 ；当允许单击时，鼠标指针变为 .

(7) 对于本游戏，要取得高分，除了应具有敏锐的观察力和快速的反应能力，还要有好运气。因为，有时一次交换位置会引起“连锁反应”得很多分，而有时无论怎样交换位置都不能形成三三相邻的关系。



图 3.41 游戏结束状态

3. 难点与提示

(1) 本题目提供了两个光标文件：hand.cur、stop.cur 用于不同时刻显示手形和禁止光标（参见 3.1.5 小节）；提供了 tick.wav、wrong.wav、ok.wav 和 alert.wav 四个声音文件用于产生走时、错误、成功和完毕等四种提示音。

(2) 本题目提供了四个图片（见图 3.42），通过 PictureClip 控件，可以用来显示动物图标和成绩数字。其中 led.bmp 中每个数字的宽和高是 18 和 24 像素。其他三个图片中每

个动物图标宽和高均为 34 像素。zoowhite.bmp 用来显示游戏区图标（背景为白色），zooblack.bmp 用来显示成绩区图标（背景为黑色），zoored.bmp 用来显示游戏结束时的游戏区图标（背景为红色）。



图 3.42 程序提供的图片文件

(3) 程序主窗体的设计可以参考图 3.43。游戏区是一个 PictureBox 图片框控件 picGrid，动物图标由 PaintPicture 方法绘制；成绩显示区也是一个图片框控件 picScore，其中的图标和数字也是使用 PaintPicture 方法画出的（“Σ”符号可以使用标签控件实现）；时间显示是由一个 ProgressBar 进度条控件实现的（参阅 3.1.7 小节）。

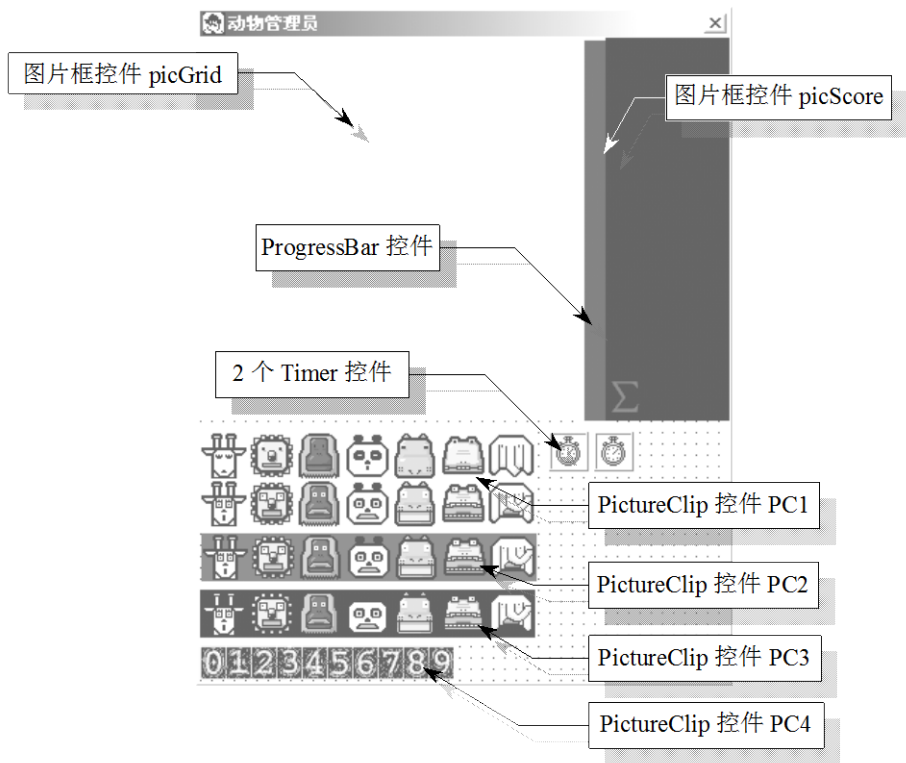


图 3.43 设计时的主窗体

如图 3.42 所示，各种图标和数字是集中地保存在大图片中的，这正好使用 PictureClip 图像裁剪控件（参阅 3.1.3 小节）。程序中共使用了 4 个 PictureClip 控件，名称分别是 PC1~PC4，分别用来管理图中所示的四个图片文件。

提示音的播放可以使用 MMControl 控件（参见 3.1.6 小节），也可以使用 API 函数

sndPlaySound 函数实现（参见 4.1.4 小节）。因为本程序的提示音较多且很密集，使用 sndPlaySound 函数的效果更好些。

（4）为了方便数据的管理与编程，可以定义以下三个全局数组。

```
Option Base 0
Public Grid(7, 7) As Integer           '游戏区的动物图标分布
Public Meet(7, 7) As Boolean           '保存形成相邻关系的位置
Public Score(7) As Integer             '已捕获的每种动物数量
```

其中，Grid 数组保存游戏区中 8×8 个位置上的图标，如果 $\text{Grid}(0,0)=0$ 表示第 1 行第 1 列（最左上角）图标为第一种动物图标。注意，这里的下标从 0 开始。 $\text{Grid}(7,7)=3$ 表示第 8 行第 8 列上（最右下角）的图标为第 4 种动物图标。因为图标共有 7 种，所以 Grid 数组中元素值的范围在 0~6 之间。

下面程序段表示了怎样随机地排列图标以及如何绘制这些图标。

```
Randomize                               '初始化随机数
For i = 0 To 7
    For j = 0 To 7
        Grid(i, j) = Int(Rnd * 7)       '随机排列动物图标
    Next
Next
For i = 0 To 7
    For j = 0 To 7                       '绘制图标
        picGrid.PaintPicture PC1.GraphicCell(Grid(i, j)), i * 34, j * 34
    Next
Next
```

逻辑型数组 Meet 用来保存哪些位置上的图标满足三三相邻的关系，元素值为 True 的位置表示满足相邻关系。每次交换相邻图标位置之后，程序都判断所有图标的相邻关系，如果有满足的，将相应位置上的 Meet 数组元素赋 True，其他元素赋 False。值为 True 元素的个数为新捕获的动物数。依据 Meet 数组的值来调整 Grid 数组元素的值，实现图标消去和下落的操作。

数组 Score 保存已捕获 7 种动物的个数和总数，共 8 个元素值。可以使用以下过程将成绩显示在成绩显示区中。

```
Private Sub ShowScore()                 '显示成绩
    Dim i As Integer
    For i = 0 To 7
        ShowNum 34, i * 34 + 5, Format(Score(i), "000")
    Next
End Sub

Private Sub ShowNum(x As Integer, y As Integer, s As String)
    Dim i As Integer
```

```

For i = 1 To Len(s)
    picScore.PaintPicture PC4.GraphicCell(CInt(Mid(s, i, 1))), _
        x + 18 * (i - 1), y
Next
End Sub

```

(5) 使用鼠标单击选择动物图标时，图标会变红（是前景变红，而不是背景变红）。本程序并未提供红色前景的图片文件，可以使用特殊的绘图模式来使图标变红。以下程序段演示了如何将第 *m* 行第 *n* 列的图标变红。

```

picGrid.DrawMode = 15                '使用模式 15
picGrid.Line (m * 34, n * 34)-(m + 1) * 34, (n + 1) * 34, vbRed, BF
picGrid.DrawMode = 13                '设回默认模式 13

```

关于绘图模式，请参阅 6.1.2 小节。

4. 更上一层楼

(1) 本程序使用了时间控制且只能是 2min，很不灵活。试着修改程序使游戏时间可调，或者使用其他方法控制时间，如总成绩达到某个数值或所有动物个数都超过某个数目时停止，等等。

(2) 增加用户登录、成绩记录、查询和排行榜等功能。