فصل اول

تصویرها، ماوس و صفحه کلید

هدف‌های رفتاری: پس از آموزش این فصل، هنرجو می‌تواند:

انواع تصویرهای گرافیکی را شناسایی کرده و آن‌ها را در برنامه‌های خود به‌کار ببرد.

- **از تابع Load Picture برای قراردادن تصویر در کنترل‌های Image و Picture Box استفاده کند.**
- **از رویدادهای صفحه کلید و ماوس در برنامه‌های خود استفاده کند.**
- **مشخصهٔ Key Preview را برای نحوهٔ دریافت ضربات صفحه کلید به وسیلهٔ فرم، مقداردهی کند.**
- **با استفاده از دستور Send Keys کلیدهایی را به برنامه ارسال کند.**

تصویرهای گرافیکی مفاهیم را به‌خوبی بیان می‌کنند. یکی از مشخصه‌های مشترک تمام قالب‌های تصویری، عمق رنگ (Color depth) است (تعداد رنگ‌هایی که تصویر از آن‌ها پشتیبانی می‌کند). عمق رنگ، اغلب با عبارت بیت به کار برده می‌شود. تعداد زیاد بیت‌ها عمق رنگ بیشتر و درنتیجه، تصویری با کیفیت بالا را ارایه می‌کنند. تصویرهایی با عمق رنگ یک بیت، تصویرهایی تک رنگ هستند (تصویرهای سیاه و سفید). تصویرهایی با عمق رنگ ۸ بیت، دارای ۲۵۶ رنگ و اغلب، تصویرهای تخت نامیده می‌شوند. تصویرهایی با عمق ۱۶ بیت، تصویرهای با کیفیت بالا نامیده می‌شوند که دارای ۶۵۵۳۶ رنگ هستند. تصویرهایی با عمق ۲۴ بیت، تصویرهای با رنگ واقعی (true Color) هستند. ویژوال بیسیک از انواع قالب‌های موجود برای تصویرها، از تصویرهای تک‌رنگ گرفته تا تصویرهای با رنگ واقعی حمایت می‌کند (جدول ۱ــ۱).

جدول ۱ــ۱ــ قالب‌های تصویری که به وسیله ویژوال بیسیک پشتیبانی می‌شوند.

نوع	پسوند	توضیحات
GIF images	.gif	Graphic Interchange Format تصویرهایی با ۲۵۶ رنگ یا کمتر
Bitmap image	.bmp	تصویرهایی با عمق رنگ 1bit، 4bit،8bit، 16bit، 24bit
Bitmap image	.dib	تصویرهایی با عمق رنگ 1bit، 4bit،8bit،16bit،24bit
Icon image	.ico	تصویرهایی با ترکیب ۱۶ رنگ، عموماً دارای اندازهٔ ۳۲ در ۳۲ پیکسل که برای نشانه‌ها استفاده می‌شوند.
Curser image	.cur	یک نوع bitmap که عموماً برای اشاره‌گر ماوس استفاده می‌شود.
Windows meta file and Enhanced Windows	.emf و .wmf	تصویرهایی گرافیکی که یک تصویر هندسی را ارایه می‌کنند (مانند خط و دایره).
RLE	.rle	یک نوع تصویر فشرده شده
JPEG image	.jpg	Joint Photographic Experts Group از تصویرهای با رنگ واقعی و تصویرهای تخت پشتیبانی می‌کند.

۱ــ۱ــ کنترل‌های تصویر و کادر تصویر

معمولاً می‌توان تصویرها را با کنترل Image یا کنترل Picture Box به نمایش درآورد. اگرچه می‌توان تصویرها را با استفاده از مشخصهٔ Picture فرم یا سایر شیءهایی که این مشخصه را دارند نیز، نمایش داد. کنترل‌های Image و Picture Box دارای مشخصه‌های شبیه به هم هستند. کنترل PictureBox دارای مشخصه‌ها (مانند Back Color) و روال رویداد (مانند Change) بیشتری نسبت به کنترل image است.

کنترل‌های کادر تصویر (picture box) و تصویر (image)، در اصل عملکرد یکسانی دارند. هر دوی آن‌ها برای نمایش تصویرها و پرونده‌های گرافیکی روی فرم هستند. اما این دو کنترل تفاوت‌هایی هم دارند :

● کنترل کادر تصویر، انعطاف‌پذیری بیشتری دارد و متدهای بیشتری را پشتیبانی می‌کند.

● کنترل تصویر کارایی بهتری دارد و در PCهای با سرعت پایین یا کند بهتر عمل می‌کند. (البته امروزه، دیگر این جنبهٔ قضیه اهمیت چندانی ندارد.)

هر دو کنترل کادر تصویر و تصویر از قالب‌های گرافیکی جدول ۱ــ۱ پشتیبانی می‌کنند.

مهم‌ترین مشخصهٔ این دو کنترل، خاصیت Picture (تصویری که کنترل باید نمایش دهد) است. برای قراردادن تصویر در این کنترل‌ها، در هنگام اجرای برنامه می‌توانید از تابع ()Load Picture استفاده کنید :

```
picPortrait.Pictuer = LoadPicture ("C: \My Photos\ Masque.wmf")
```

دقت کنید که نمی‌توان به‌طور مستقیم یک پرونده تصویری را به خاصیت Picture نسبت داد. تابع ()Load Picture یکی از مهم‌ترین توابع کار با پرونده گرافیکی است. شکل کلی این تابع چنین است :

```
LoadPictuer ([GraphicFileName],[,varSize],[,varColorDepth],[varX,varY])
```

توجه کنید که حتی نام پرونده هم در این تابع اختیاری است و اگر از این تابع بدون مشخص‌کردن نام پرونده استفاده کنید، ویژوال بیسیک کنترل تصویر را پاک خواهد کرد.

باید توجه داشت که متد cls، فقط عبارت‌های نوشته‌شده یا نقاط رسم شده در کنترل تصویر را پاک می‌کند ولی این تابع، می‌تواند عکس درون کنترل تصویر را پاک کند.

آرگومان varSize می‌تواند مقادیر مختلفی بگیرد که آن‌ها را در جدول ۲ــ۱ مشاهده می‌کنید. این آرگومان که اندازهٔ تصویر برای پرونده نشانه یا اشاره‌گر را تعیین می‌کند، یکی از آرگومان‌های مهم این تابع است.

جدول ۲ــ۱ــ مقادیر آرگومان varSize تابع ()Load Picture (فقط برای پرونده‌های نشانه و اشاره‌گر)

ثابت نام‌دار	مقدار	مفهوم
VbLPSmall	۰	نشانه‌های کوچک
VbLPLarge	۱	نشانه‌های بزرگ
VbLPSmallShell	۲	از تنظیمات Display| Appearance مرکز کنترل ویندوز استفاده می‌کند.
VbLPLargeShell	۳	از تنظیمات Display	Appearance مرکز کنترل ویندوز استفاده می‌کند.
VbLPCustom	۴	اندازهٔ نشانه یا آرگومان‌های varX و varY تعیین خواهد شد.

در جدول ۳ــ۱ هم مقادیر آرگومان اختیاری varColor Depth را مشاهده می‌کنید.

جدول ۳ــ۱ مقادیر آرگومان varColorDepth تابع Load Picture() (فقط برای پرونده‌های نشانه یا اشاره‌گر)

ثابت نام‌دار	مقدار	مفهوم
VbLPDefault	۰	بهترین انطباق
VbLPMonochrome	۱	۲ رنگ
VbLPVGAColor	۲	۱۶ رنگ
VbLPColor	۳	۲۵۶ رنگ

کنترل‌های کادر تصویر و تصویر حتی اگر به یک اندازه باشند و تصویر مشابهی را نمایش دهند، به‌طور متفاوتی عکس‌العمل نشان می‌دهند. در کنترل تصویر، قبل از تنظیم کردن خواص Width و Height، باید خاصیت Stretch با True مقداردهی شده باشد. در غیراین صورت، اندازهٔ کنترل به‌طور خودکار به اندازهٔ تصویر موجود در آن درخواهد آمد. اما در مورد کنترل کادر تصویر، فقط بخشی از تصویر که به اندازهٔ کنترل است، نمایش می‌یابد. به عبارت دیگر، کادر تصویر قسمتی از تصویر را نمایش می‌دهد، ولی کنترل تصویر خود به اندازهٔ تصویر درمی‌آید (مگر آن که خاصیت Stretch با True تنظیم شده باشد).

حتی فرم‌ها هم خاصیت Picture را دارند و می‌توانید با تابع Load Picture() تصویرها را به‌طور مستقیم روی فرم قرار دهید. شکل ۲ــ۱ فرمی را که یک تصویر در آن قرار داده شده است، نشان می‌دهد.

تابع load Picture() چه چیزی برمی‌گرداند؟ واضح است، تصویری که نام پرونده آن داده شده است. مشاهده می‌کنید که مقدار برگشتی تابع Load Picture() به خاصیت Picture کنترل تصویر نسبت داده می‌شود، زیرا بدون انجام این کار، کنترل مزبور نمی‌تواند تصویر را نمایش دهد.

هنگامی که ویژوال بیسیک با تابع Load Picture() برخورد کند، وجود یا وجود نداشتن پروندهٔ مشخص شده را بررسی می‌کند (اگر پروندهٔ مزبور روی شبکه باشد، ویژوال بیسیک باید بررسی کند که آیا شما حق استفاده از آن را دارید یا خیر). در پایان هم پروندهٔ تصویر را خوانده و آن را در کنترل تصویر قرار می‌دهد.

شکل ۱ــ۱ــ تصویرها می‌توانند به‌طور مستقیم روی فرم قرار گیرند.

برنامهٔ ۱ــ۱ تصویری را به داخل یک کادر تصویر بار می‌کند. سپس با فشار دکمهٔ Inverse، یک تصویر معکوس از تصویر اصلی ایجاد می‌شود. همچنین برنامه، تصویر معکوس شده را ذخیره می‌کند. روال cmdInvert_Click متد PaintPicture را فراخوانی می‌کند تا معکوس تصویر در کادر تصویر به نمایش درآید. آرگومان اول متد PaintPicture مشخص می‌کند که تصویر رسم شود. آرگومان دوم و سوم مشخص‌کنندهٔ مختصات x و y هستند و تعیین می‌کنند که تصویر در گوشهٔ سمت چپ بالا به نمایش درآید. PaintPicture دارای هفت آرگومان اختیاری برای ایجاد جلوه‌های گوناگون است. تعدادی ازاین آرگومان‌ها بر روی تصویرهای نقش بیتی تأثیر دارند. آرگومان آخر یک مقدار Long است که نحوهٔ انجام عمل را مشخص می‌کند. ثابت vbDstInvert، یک عمل معکوس‌کننده بر روی تصویر نقش بیتی انجام می‌دهد.

برنامهٔ ۱ــ۱ــ معکوس‌کردن یک تصویر

```
1. Private Sub Form_Load ()
2. PicPicture.Picture = Load Picture ("e: \image\cho9\cool.bmp")
3. End Sub
4. Private sub cmdInvert _ Click ( )
5. picPicture.PaintPicture picPicture.Picture, 0, 0, , , , , , , vbDstInvert
6. Save Picture picPicture, "d: \images\cho9\" & "Cool _ invertse. bmp"
7. End Sub
```

متد Save Picture تصویر را بر روی دیسک ذخیره می‌کند. این متد دو آرگومان دریافت می‌کند : یک تصویر و یک رشته، محل و نام پرونده را مشخص می‌کند (شکل ۲ــ۱). (توجه : پرونده تصویری مورد نظر باید در مسیر باشد.)

شکل ۲ــ۱

۲ــ۱ــ ماوس و صفحه کلید

یکی از اساسی‌ترین اصول برنامه‌های ویندوز این است که آن‌ها نسبت به عملیات ماوس عکس‌العمل نشان می‌دهند. هرگاه کاربر عملی با ماوس انجام دهد، ویندوز رویدادی را به برنامه می‌فرستد. هنگام برنامه‌نویسی باید این رویدادها را بررسی کرده و در پاسخ به هرکدام، اقدام مناسب را انجام دهید. البته بعضی از رویدادهای ماوس (به عنوان مثال، کلیک‌کردن روی کادر علامت یا دکمهٔ انتخاب) نیازی به اقدام برنامه‌نویس ندارند، ولی برنامه‌نویس باید رویدادهای دیگر را کنترل کند. هنگامی که کاربر شیئی را روی فرم برنامه جابه‌جا می‌کند یا اطلاعاتی را کپی کرده و در نقطه‌ای دیگر می‌چسباند، خود ویژوال بیسیک بر رویدادها و اعمال ماوس نظارت خواهد کرد.

البته این یکی از استانداردهای ویندوز است که یک برنامه حتی بدون ماوس و فقط به کمک صفحه کلید هم باید بتواند تمام کارایی خود را حفظ کند. با این حال، طبیعت برخی برنامه‌ها به گونه‌ای است که بدون ماوس نمی‌توانند به درستی عمل کنند؛ برنامه‌های طراحی و نقاشی از این قبیل هستند.

کاربر به وسیلهٔ ماوس و صفحه کلید با برنامه‌های ویژوال بیسیک ارتباط برقرار می‌کند. از ماوس برای کلیک‌کردن روی دکمه‌ها، جابه‌جاکردن پنجره‌ها، تغییر اندازهٔ پنجره‌ها، ترسیم نقاط و شکل‌ها و

سایر عملیات استفاده می‌شود. کنترل‌های ویژوال بیسیک می‌توانند کلیک، دوبار کلیک، جابه‌جایی ماوس، و فشار داده‌شدن یا رهاشدن دکمهٔ ماوس را تشخیص دهند. مانند سایر رویدادها، برنامه نویس می‌تواند برای رویدادهای ماوس نیز کد بنویسد. برنامه می‌تواند بررسی کند که کدام دکمهٔ ماوس (چپ، وسط یا راست) فشار داده شده است. ویژوال بیسیک از مفهوم کشیدن و رها کردن (drag and drop) به وسیلهٔ ماوس نیز پشتیبانی می‌کند.

۱ــ۲ــ۱ــ رویدادهای ماوس : ماوس می‌تواند رویدادهای مختلفی تولید کند :

- جابه‌جایی ماوس
- کلیک (click)
- دوبار کلیک (double – click)
- کلیک راست (right – click)
- عملیات کشیدن ــ رها کردن (Drag & Drop)

۲ــ۲ــ۱ــ تغییر شکل اشاره‌گر ماوس : هنگامی که کاربر، ماوس خود را جابه‌جا می‌کند، اشاره‌گر ماوس (Pointer) روی صفحه جابه‌جا می‌شود. در اغلب برنامه‌ها، هنگامی که کاربر کار خاصی با ماوس انجام می‌دهد یا در ناحیهٔ خاصی حرکت می‌کند، اشاره‌گر ماوس تغییر شکل می‌دهد. به‌عنوان مثال، تبدیل اشاره‌گر ماوس به یک ساعت شنی نشان می‌دهد که کار خاصی در حال انجام است و کاربر باید صبر کند.

شکل اشاره‌گر ماوس، تحت کنترل برنامه‌نویس است. در جدول ۴ــ۱ شکل‌های مختلف اشاره‌گر ماوس را مشاهده می‌کنید. برای تغییر شکل اشاره‌گر، هنگامی که ماوس روی یک کنترل خاص است، باید از مشخصهٔ MousePointer آن کنترل استفاده کنید. تقریباً تمام کنترل‌های ویژوال بیسیک مشخصهٔ MousePointer را دارند.

مشخصهٔ MousePointer فرم یا کنترل، شکل اشاره‌گر ماوس را هنگامی که اشاره‌گر وارد ناحیهٔ کنترل می‌شود، تعیین می‌کند. به این منظور، ویژوال بیسیک دارای ثابت‌هایی است که شکل اشاره‌گر ماوس را تغییر می‌دهند (جدول ۴ــ۱). این تغییر شکل در واقع عمل خاصی انجام نمی‌دهد، ولی می‌تواند از لحاظ بصری بیان‌کنندهٔ انجام عمل خاصی باشد. به‌عنوان مثال، در نرم‌افزار وُرد اشاره گر ماوس به شکل I بر این نکته دلالت دارد که کاربر می‌تواند متنی را در داخل سند با کلیک کردن در آن نقطه وارد کند.

ویندوز امکان استفاده از تصویرهای متحرک، در اشاره‌گر ماوس را فراهم می‌کند. مکان‌نماهای متحرک دارای پسوند ANI هستند. مکان‌نماهای متحرک را نمی‌توان در ویژوال بیسیک به‌کار برد.

جدول ۴ــ۱ــ ثابت‌های MousePointer

مقدار	ثابت‌ها	مفهوم
۰	vbDefault	شکل پیش‌فرض اشاره‌گر
۱	vbArrow	علامت معمولی اشاره‌گر 
۲	vbCrosshair	علامت +
۳	vbIbeam	علامت I
۵	vbSizePointer	
۶	vbSizeNESW	
۷	vbSizeNS	
۸	vbSizeNWSE	
۹	vbSizeWE	
۱۰	vbUpArrow	
۱۱	vbHourglass	
۱۲	vbNoDrop	
۱۳	vbArrowHourglass	
۱۴	vbArrowQuestion	
۱۵	vbSizeAll	
۹۹	vbCustom	شکلی که به خاصیت Mouse Icon اشاره می‌کند.

نکته

در MousePointer استفاده از مکان‌نماهای متحرک امکان ندارد.

اشاره‌گر ماوس یک نشانه ۱۶×۱۶ (با پسوند ICO) است که خودتان هم می‌توانید آن را بسازید. برای نمایش این اشاره‌گر، ابتدا باید آن را در خاصیت MouseIcon قرار داده و سپس خاصیت

MousePointer را با Custom–99 مقداردهی کنید. تا زمانی که اشاره‌گر را دوباره تغییر نداده‌اید، این شکل به‌عنوان اشاره‌گر انجام وظیفه خواهد کرد.

وظیفهٔ ایجاد رویدادهای ماوس و ارسال آن‌ها به برنامه، بر عهدهٔ سیستم عامل است. اگر برنامه‌نویس روالی برای این رویدادها ننوشته باشد، برنامه آن‌ها را نادیده خواهد گرفت. در جدول ۵ــ۱ مشاهده می‌کنید که ماوس چه رویدادهایی می‌تواند داشته باشد.

جدول ۵ــ۱ــ رویدادهای ماوس

رویداد	مفهوم
Click	کاربر یکی از دکمه‌های ماوس را کلیک کرده است.
DblClick	کاربر یکی از دکمه‌های ماوس را دو بار کلیک کرده است.
MouseDown	کاربر دکمهٔ ماوس را فشار داده و نگه داشته است.
MouseMove	کاربر ماوس را جابه‌جا کرده است.
MouseUp	کاربر دکمهٔ ماوس را رها کرده است.

تمام رویدادهای ماوس به کنترل‌ها وابسته‌اند و تقریباً تمام کنترل‌ها (و فرم) دارای رویدادهای ماوس هستند، بجز کنترل‌هایی که شکل ظاهری ندارند (مثل Timer). به‌عنوان مثال، اگر فرمی به نام frm Test داشته باشید، کلیک‌کردن روی آن سبب ایجاد رویداد frmTest_Click خواهد شد. در برخی رویدادها لازم است برنامه‌نویس بداند که کاربر کدام دکمهٔ ماوس را کلیک کرده است. این کار فقط در رویدادهای MouseDown و MouseUp امکان‌پذیر است.

اما آیا دوبار کلیک یک رویداد مستقل است یا دو رویداد کلیک پشت سرهم؟ پاسخ این پرسش به دقت کاربر در اجرای دو بار کلیک بستگی دارد. ترتیب رویدادهای ماوس از نظر ایجاد آن‌ها به وسیلهٔ ویندوز چنین است :

۱ــ MouseDown

۲ــ MouseUp

۳ــ Click

۴ــ Dblclick

۵ــ MouseUp

بررسی کنید

دلیل تکراری بودن موارد ۲ و ۵ چیست؟

یعنی، ابتدا یک رویداد MouseDown روی می‌دهد، سپس یک MouseUp و به دنبال آن Click. هنگامی که کاربر دو بار کلیک کند، رویدادهای DblClick و MouseUp به دنبال آن‌ها روی خواهند داد.

رویدادهای MouseDown، MouseMove و MouseUp چهار آرگومان ورودی می‌گیرند :

- **intButton** : دکمه‌ای که فشار داده شده : ۱ برای دکمهٔ چپ، ۲ برای دکمهٔ راست و ۴ برای دکمهٔ وسط (اگر وجود داشته باشد).
- **intShift** : فشار داده شدن دکمه‌های Ctrl، Alt و Shift (همزمان با رویداد) را منعکس می‌کند (۴ برای Alt، ۲ برای Ctrl و ۱ برای Shift).
- **sngX** : مختص افقی اشاره‌گر ماوس در هنگام وقوع رویداد.
- **sngY** : مختص عمودی اشاره‌گر ماوس در هنگام وقوع رویداد.

ویژوال بیسیک فقط بعد از حرکت ماوس به میزان ۱۰ تا ۱۵ twip [۱] (که مقداری بسیار جزئی و کوچک است)، یک رویداد MouseMove تولید می‌کند و برای هر twip این رویداد را ایجاد نخواهد کرد. شکل کلی رویداد MouseDown چنین است :

```
Private Sub imgMouse _ MouseDown (intButton As Integer, intShift As Integer,
sngX As Single, sng Y As Single)
```

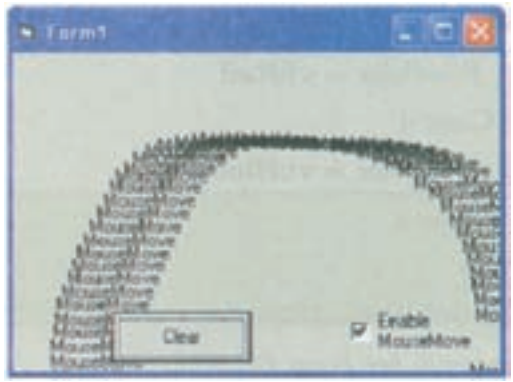
آرگومان‌های sngX و sngY مکان دقیق ماوس را هنگام ایجاد این رویداد برمی‌گردانند. آرگومان intButton (که برای ماوس‌های سه‌دکمه‌ای می‌تواند ۱، ۲ یا ۴ باشد)، کلید فشار داده شده را برمی‌گرداند. البته در اکثر موارد دکمهٔ فشار داده شده چندان مهم نیست، ولی اگر می‌خواهید برنامه با توجه به این که چه دکمه‌ای فشار داده شده است، عکس‌العمل‌های متفاوتی از خود بروز دهد، جای تعیین آن همین رویداد MouseDown است. برنامهٔ ۲ــ۱ برنامه‌ای را نشان می‌دهد که در آن همزمانی کلیدهای Alt، Ctrl یا Shift با رویداد MouseDown بررسی شده است.

۱ــ 1 inch = 2.54 cm = 72 points = 1440 twips

برنامهٔ ۲ــ۱ــ تعیین کلید زده شده همزمان با یک رویداد ماوس

```
1. Private Sub imgMouse_MouseDown (intButton As Integer, intShift As
   Integer, sngX As Single, sngY As Single)
2. Dim intShifState As Integer
3. intsShiftState = intShift And 7      'Special bitwise And
4. Select Case intShiftState
5. Case 1
6.         'Code for Shift combinations
7. Case 2
8.         'Code for Ctrl combinations
9. Case 3
10.    'Code for shift + Alt combinations
11. Case 4
12.    'Code for Alt combinations
13. Case 5
14.        'Code for shift + Alt combinations
15. Case 6
16.        'Code for Ctrl + Alt combinations
17. Case 7
18.        'Code for shift + Ctrl + Alt combinations
19. End Select
20. End Sub
```

در خط ۳ برای بررسی کلید زده شده از And با عدد ۷ (۱۱۱ دودویی) استفاده شده است.



شکل ۳ــ۱

برنامهٔ ۳ــ۱ نحوهٔ استفاده از رویدادهای ماوس را نشان می‌دهد (شکل ۳ــ۱).

برنامهٔ ۳ــ۱ــ استفاده از رویدادهای ماوس

```
1. 'Demonstrating mouse events
2. Option Explicit  'General declaration
3. Private Sub  Form_Load ( )
4.         Call randomize     'Randomize
5. End Sub
6. Private Sub cmdClear_Click ( )
7.         Call Cls  'Clear Form
8. End Sub
9. Private Sub  Form_Click ( )
10.        'Randomly Set Form ForeColor
11.        Select Case  (1+Int (Rnd ( )* 4))
12.          Case 1
13.           ForeColor = vbBlack
14.          Case 2
15.           ForeColor = vbMagenta
16.          Case 3
```

```
17.            ForeColor = vbRed
18.          Case 4
19.            ForeColor = vbBlue
20.   End Select
21. End Sub
22. Private Sub Form_Dblclick ( )
23.       'Randomly Set Form BackColor
24.       Select Case (1 + Int (Rnd( ) )* 4)
25.          Case 1
26.            BackColor = vbWhite
27.          Case 2
28.            BackColor = vbYellow
29.          Case 3
30.            BackColor = vbGreen
31.          Case 4
32.            BackColor = vbCyan
33.    End Select
34.    'Changing ChkMove  BackColor  to Form's  BackColor
35.    chkMove. BackColor = BackColor
36. End Sub
37. Private Sub Form_Mouse Down (Button As Integer,
38.        Shift As Integer, X As Single, Y As single
39.        CurrentX = X  'Set x Coordinate
40.        CurrentY = Y 'Set y Coordinate
41.        Print "MouseDown"
42.       End Sub
43.      Private Sub Form_MouseUp (Button As Integer,
```

```
44.     Shift as Integer, X As Single, Y As Single)
45.     'Reverse Coordinates
46.     Current X = Y
47.     Current Y = X
48.     Print "MouseUp"
49. End Sub
50. Private Sub Form_MouseMove (Button As Integer,
51.     Shift As Integer, X As Single, Y as Single)
52.     'If checked enable Printing Operations
53.     If chkMove. Value = 1 Then
54.       CurrentX = X
55.       Current Y  = Y
56.       Print "MouseMove"
57.    End If
58. End Sub
```

روال Form_MouseDown هنگامی فراخوانی می‌شود که دکمهٔ سمت چپ یا راست ماوس فشار داده شده باشد. مشخصه‌های CurrentX و CurrentY فرم در حالتی که MouseDown رخ دهد با مختصات X و Y تنظیم می‌شوند. این مشخصه‌ها تعیین می‌کنند که متد Print در زمان اجرا، در کدام محل اعمال شود. عبارت "MouseDown" در مکانی که MouseDown رخ داده است، چاپ می‌شود. هنگام رهاشدن دکمهٔ ماوس، روال Form_MouseUp فراخوانی خواهد شد. CurrentX با Y ر CurrentY با X مقداردهی شده است. به این ترتیب عبارت "MouseUP" بر روی "MouseDown" چاپ نمی‌شود.

با کلیک‌کردن روی فرم، روال Form_Click فراخوانی شده و رنگ رویه فرم را به سیاه، ارغوانی، قرمز یا آبی تغییر می‌دهد. با دو بار کلیک سریع بر روی فرم، روال Form_DblClick فراخوانی شده و رنگ زمینه به صورت تصادفی به رنگ های سفید، زرد، سبز یا فیروزه‌ای تغییر می کند. با حرکت‌دادن ماوس بر روی فرم، روال Form_MouseMove فراخوانی می‌شود.

اگر کادر علامت chkMove علامت‌دار شده باشد، عبارت "MouseMove" در موقعیتی که

ماوس قرار دارد، به چاپ می‌رسد. در غیر این صورت، چیزی چاپ نمی‌شود.

۳ــ۲ــ۱ــ دکمه‌های ماوس

آرگومان Button در روال‌های MouseDown، MouseUp و MouseMove تعیین‌کنندهٔ دکمه‌ای از ماوس است که فشار داده شده است. ماوس می‌تواند یک، دو یا سه دکمه داشته باشد.

جدول ۶ــ۱ لیستی از مقادیر ثابت که با آرگومان Button در ارتباط هستند را ارایه می‌کند.

جدول ۶ــ۱ــ ثابت‌های دکمهٔ ماوس

ثابت‌ها	مقدار	توضیحات
vbRightButton	۲	دکمهٔ سمت راست ماوس
vbLeftButton	۱	دکمهٔ سمت چپ ماوس
vbMiddleButton	۴	دکمهٔ وسط ماوس

برنامهٔ مثال زیر، تعیین می‌کند کدام‌یک از دکمه‌های ماوس فشار داده شده است.

مثال ۱ــ۱

فرمی طراحی کنید که دارای یک کنترل Image به نام imgImage باشد، سپس کد زیر را در پنجرهٔ کد این فرم وارد کنید. مسیر پرونده‌ها را متناسب با مسیر آن‌ها در رایانه خودتان انتخاب کنید.

```
1. 'Mouse buttons
2. Options Explicit  'General declaration
3. Private Sub  Form_Load ( )
4.          imgImage.Picture = LoadPicture ("d: \mouse0.gif")
5. End Sub
6. Private Sub Form_MouseDown (Button As Integer, Shift As Integer, X As
Single, Y As Single)
7.    imgImage.Picture = LoadPicture ("d: \mouse" & Button &".gif")
8. End Sub
```

```
9. Private Sub Form_MouseUp (Button As Integer, Shift As Integer, X As
Single, Y As Single)
10.    imgImage.Picture = LoadPicture ("d: \mouse0.gif")
11. End Sub
12. Private Sub imgImage_MouseDown(Button As Integer, Shift As Integer,
X As Single, Y as Single)
13.    imgImage. Picture = LoadPicture ("d: \mouse"& Button & ".gif")
14. End Sub
15. Private Sub imgImage_MouseUp (Button As Integer, Shift As Integer, X
As Single, Y As Single)
16. imgImage.Picture = LoadPicture ("d: \mouse0.gif")
17. End Sub
```

۴ــ۲ــ۱ــ کلیدهای Shift، Ctrl و Alt : می‌توان وضعیت کلیدهای Shift، Ctrl و Alt را در هنگامی که رویدادهایMouseDown، MouseMove یا MouseUp رخ می‌دهند، تعیین کرد. مقداری که نشان‌دهنده وضعیت این کلیدهاست در متغیری به نام Shift ذخیره می‌شود. جدول ۷ــ۱ شامل ثابت‌هایی است که با متغیر Shift در ارتباط هستند. ثابت‌های Shift نیز مانند ثابت‌های Button، می‌توانند حالت ترکیبی داشته باشند.

جدول ۷ــ۱ــ ثابت‌هایی که با متغیر Shift در ارتباط هستند.

ثابت‌ها	مقدار	توضیحات
vbShiftMask	۱	وضعیت کلید Shift
vbCtrlMask	۲	وضعیت کلید Ctrl
vbAltMask	۴	وضعیت کلید Alt

۵ــ۲ــ۱ــ کشیدن و رهاکردن : وقتی کاربر یک شیء را (روی فرم) از نقطه‌ای به نقطهٔ

دیگر می‌کشد، برنامه باید از آن مطلع باشد! کشیدن ــ رهاکردن (drag&drop) عملیاتی است که طی آن کاربر روی یک شیء کلیک کرده، دکمهٔ ماوس را نگه می‌دارد و آن شیء را به نقطه‌ای دیگر می‌برد. به کمک ویندوز، برنامه‌نویسی کشیدن ــ رهاکردن نسبتاً ساده است، چون ویندوز تمام اطلاعات دربارهٔ این فرایند را در اختیار برنامه‌نویس قرار خواهد داد. ویژوال بیسیک دو نوع عملیات کشیدن ــ رهاکردن دارد :

- کشیدن ــ رهاکردن خودکار
- کشیدن ــ رهاکردن دستی

روش اول :

انجام این روش، از طریق خواص کنترل صورت می‌گیرد. تقریباً تمام کنترل‌های ویژوال بیسیک دارای خاصیت DragMode هستند. این خاصیت امکان می‌دهد تا کاربر یک کنترل را با ماوس جابه‌جا کند (وقتی کنترل در حال حرکت است، ویژوال بیسیک فقط قاب آن را نمایش خواهد داد). این وظیفهٔ برنامه‌نویس است که وقتی کاربر کنترل را در نقطه‌ای رها کرد، آن کنترل را به نقطهٔ موردنظر منتقل کند. در حالت خودکار، با اینکه کاربر می‌تواند قاب کنترل را حرکت دهد، ولی انتقال آن بر عهدهٔ برنامه‌نویس است.

رهاشدن کنترل‌ها بر عهدهٔ رویداد DragDrop فرم است. برای اجرای عملیات، فقط باید خاصیت DragMode کنترل با مقدار Automatic–1 مقداردهی شود. روال Form_DragDrop()مسئول نیمهٔ دوم عملیات و انتقال کنترل به محل جدید است.

با آن که ویژوال بیسیک در حالت عادی، هنگام کشیدن و انتقال کنترل‌ها فقط قاب آن‌ها را نشان می‌دهد اما این وضع قابل اصلاح است؛ خاصیت DragIcon به نشانه‌ای اشاره می‌کند که هنگام کشیدن کنترل دیده خواهد شد. بعد از پایان عملیات کشیدن، انتقال کنترل به نقطهٔ جدید در روال Form_DragDrop() صورت خواهد گرفت. به کد زیر که عملیات در آن تکمیل می‌شود، توجه کنید :

```
Private Sub Form_DragDrop (Source As Control, X As Single, Y As Single)
    Source.Move X,Y        'Move to the dropped location
End Sub
```

برای انتقال کنترل از محل اولیه به محل جدید از متد Move استفاده شده است.

نکته

وقتی کاربر کنترلی را از روی کنترل دیگر عبور می‌دهد، رویداد DragOver روی خواهد داد. اگر می‌خواهید کاربر را از رهاکردن کنترلی که در حال کشیدن آن است، روی کنترل مزبور منع کنید، در روال رویداد DragOver آن شکل اشاره‌گر را به vbNoDrop تغییر دهید. رویداد DragOver دارای چهار آرگومان است.

Private Sub Form_DragOver (source As Control، x As Single، y As Single، state As Integer)

- کنترل (source)
- مختص افقی ماوس (x)
- مختص عمودی ماوس (y)
- وضعیت کشیدن: ۰ (کنترل وارد محدودهٔ کنترل زیرین می‌شود)، ۱ (کنترل محدودهٔ کنترل زیرین را ترک می‌کند)، ۲ (کنترل در حال عبور از روی کنترل زیرین است). (State)

روش دوم :

- خاصیت DragMode باید با 0–Manual مقداردهی شود.
- کنترل به رویداد MouseDown پاسخ می‌دهد (روش خودکار چنین نیست) و مکان اولیهٔ کنترل می‌تواند ثبت شود.
- عملیات کشیدن در روال رویداد MouseDown شروع خواهد شد.

برای انجام عملیات کشیدن ــ رهاکردن در روال رویداد MouseDown می‌توانید از متد خاصی به نام Drag استفاده کنید. کد زیر نحوهٔ کشیدن یک کنترل تصویر را در حالت دستی نشان می‌دهد :

```
Private Sub imgMouse_MouseDown(Button As Integer, Shift As Integer, X
As Single, Y As Single)
    txtMouse. Text = "Click over the image at "& X &", "& Y imgMouse.Drag
End Sub
```

متد Drag سبب شروع عملیات کشیدن ــ رهاکردن خواهد شد. بدون این متد، روال

() MouseDown قادر به انجام عملیات نخواهد بود. دلیل عمدهٔ استفاده از عملیات دستی، محدود کردن حوزهٔ عملیات مطابق خواستهٔ برنامه‌نویس است (جدول ۸ــ۱).

جدول ۸ــ۱ــ ثابت‌های مورد استفاده در متد Drag

ثابت‌ها	مقدار	توضیحات
vbCancelDrag	۰	عمل کشیدن و رهاکردن لغو می‌شود. روال DragDrop فراخوانی نمی‌شود.
vbBeginDrag	۱	عمل کشیدن و رهاکردن شروع می‌شود. روال DragDrop فراخوانی می‌شود.
vbEndDrag	۲	عمل کشیدن و رهاکردن خاتمه می‌یابد. روال DragDrop فراخوانی می‌شود.

مشخصهٔ DragIcon در مدتی که عمل کشیدن و رهاکردن در حال انجام است تصویری به نمایش درمی‌آورد.

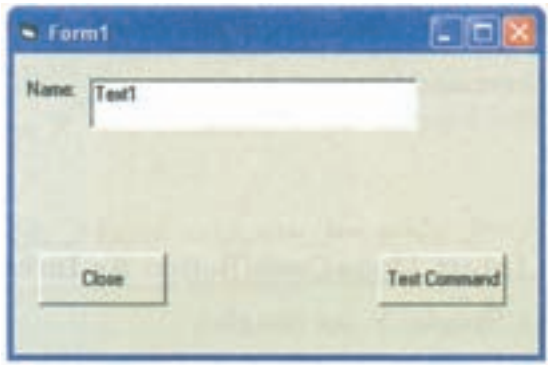
مثال ۲ــ۱

برنامهٔ زیر دارای یک کادر متن به نام Text1، یک برچسب به نام labell و یک کلید به نام cmdTestCommand است که به روش دستی قابل کشیدن هستند.

مراحل کار :

۱ــ برنامهٔ ویژوال بیسیک را اجرا کرده و پروژهٔ جدیدی ایجاد کنید.

۲ــ فرمی به‌صورت شکل ۴ــ۱ طراحی کنید و کد مربوط به دکمهٔ Close آن را نیز بنویسید.



شکل ۴ــ۱

۳ــ روی فرم دو بار کلیک کرده و در رویداد DragDrop فرم، دستورهای زیر را بنویسید :

```
Private Sub Form_DragDrop (Source As Control, X As Single, Y As Single)
        Source.Left = X
        Source.Top = Y
    End Sub
```

توجه

در این رویداد، آرگومان Source یک متغیر است و نام همان کنترلی را که عمل درگ روی آن رخ داده است، برمی‌گرداند. X و Y مختصات محلی است که عمل Drop در آن نقطه رخ داده است. درنتیجه به هنگام Drop دو فرمان فوق، کنترل مورد نظر را در محل رهاشدن ماوس قرار می‌دهند. یعنی گوشهٔ بالای سمت چپ کنترل در مختصات X و Y قرار می‌گیرد.

۴ــ رویدادهای MouseDown سه کنترل دیگر را نیز به‌صورت زیر بنویسید :

```
Private Sub CmdTestCommand_MouseDown(Button As Integer, Shift As
Integer, X As Single, Y As Single)
        Text1 = "Command Run Draging"
        CmdTestCommand.Drag
End Sub

Private Sub Labell_MouseDown(Button As Integer, Shift As Integer, X As
Single, Y As Single)
        Text1 = "Label Draging"
        Labell.Drag
End Sub

Private Sub Text1_MouseDown (Button As Integer, Shift As Integer, X As
Single, Y As Single)
```

```
        Text1 = "TextBox Draging"
        Text1.Drag
End Sub
```

۵ــ برنامه را اجرا کنید و سعی کنید این سه کنترل را جابه‌جا کنید.

مشاهده خواهید کرد که این سه کنترل جابجا می‌شوند ولی پس از رهاکردن دکمهٔ ماوس، گوشهٔ چپ و بالای کنترل در محل رهاشدن ماوس قرار می‌گیرد. برای رفع این اشکال، برنامه را به‌صورت زیر تغییر دهید :

```
Option Explicit
Dim intOldx As Integer, intOldy As Integer
Private Sub CmdClose_Click ()
        End
End Sub

Private Sub CmdTestCommand_MouseDown (Button As Integer, Shift As
Integer, X As Single, Y As Single)
        Text1 = "Command Run Draging"
        CmdTestCommand. Drag
        intOldx = X
        intOldy = Y
End Sub

Private Sub Form_DragDrop(Source As Control, X As Single, Y As Single)
        Source.Left = X – intOldx
        Source.Top = Y – intOldy
        End Sub

Private Sub Labell_MouseDown (Button As Integer, Shift As Integer, X As
```

```
Single, Y As Single)
        Text1 = "Label Draging"
        Labell.Drag
        intOldx = X
        intOldy = Y
End Sub

Private Sub Text1_MouseDown (Button As Integer, Shift As Integer, X As
Single, Y As Single)
        Text1 = "TextBox Draging"
        Text1.Drag
        intOldx = X
        intOldy = Y
End Sub
```

توجه

- متغیرهایی که در بخش General تعریف شوند، در تمام رویدادها قابل دسترس هستند.
- در هر رویداد MouseDown، در انتها، مقادیر X و Y (همان مختصات محلی از کنترل، که ماوس در آن قرار دارد) در دو متغیر intOldx و intOldy ذخیره می‌شود تا در لحظهٔ Drop، این مقادیر از مقادیر X و Y نقطهٔ جدید کم شده و درنتیجه، کنترل به‌درستی سر جای واقعی خود قرار گیرد.

۶ــ برنامه را اجرا کنید و سعی کنید این سه کنترل را دوباره جابه‌جا کنید. مشاهده خواهید کرد که این سه کنترل به درستی جابه‌جا می‌شوند.

درصورتی که بخواهید هنگام کشیدن، شکل اشاره‌گر ماوس عوض شود، لازم است در پایان رویدادهای MouseDown هر سه کنترل، فرمان

نام کنترل.MousePointer=vbSizeAll

را اضافه کنید و برای بازگشت شکل اشاره‌گر به حالت اولیه لازم است در انتهای رویداد Form_DragDrop، فرمان Source.MousePointer=vbArrow را اضافه کنید.

لیست برنامه پس از انجام تغییرات، به‌صورت زیر است :

```
Option Explicit
Dim intOldx As Integer, intOldy As Integer
Private Sub CmdClose_Click()
    End
End Sub
Private Sub Form_DragDrop(Source As Control, X As Single, Y As Single)
        Source.Left = X – intOldx
        Source.Top = Y – intOldy
        Source.MousePointer = vbArrow
End Sub
Private Sub CmdTestCommand_MouseDown (Button As Integer, Shift As
Integer, X As Single, Y As Single)
Text1 = "Command Run Draging"
CmdTestCommand.Drag
intOldx = X
intOldy = Y
CmdTestCommand. MousePointer = vbSizeAll
End Sub
Private Sub Labell_MouseDown (Button As Integer, Shift As Integer, X As
Single, Y As Single)
Text1 = "Label Draging"
Labell.Drag
intOldx = X
intOldy = Y
```

```
Label1.MousePointer = vbSizeAll
End Sub
Private Sub Text1_MouseDown (Button As Integer, Shift As Integer, X As
Single, Y As Single)
Text1 = "TextBox Draging"
Text1.Drag
intOldx = X
intOldy = Y
Text1.MousePointer = vbSizeAll
End Sub
```

اگر بخواهید از کشیدن خودکار استفاده کنید، لازم است به نکات زیر توجه کنید :

- مشخصهٔ DragMode مربوط به هر کنترل باید روی عدد ۱ یا vbAutomatic قرار گیرد.
- در حالت کشیدن خودکار، رویدادهای MouseDown و MouseUp غیرفعال می‌شوند و فقط رویداد MouseMove قابل استفاده است.
- نیازی به متد Drag نیست و این عمل به‌صورت خودکار انجام می‌شود.
- از مشخصهٔ DragIcon می‌توان برای تعیین شکل اشاره‌گر ماوس به هنگام کشیدن، استفاده کرد.
- رویداد DragDrop هر کنترل، هنگامی فراخوانی می‌شود که ماوس روی آن رها شود. (معادل رویداد MouseUp).

۳ــ۱ــ رویدادهای کلید

هنگامی که کاربر با صفحه کلید کار می‌کند، رویدادهای کلید (key events) ایجاد می‌شوند. با توجه به این‌که فرم یا کنترل فعال باشد پردازش این رویه به‌عهدهٔ آن شیء است. ویژوال بیسیک دارای سه روال رویدادهای کلید است : Keypress، KeyDown و KeyUp. روال‌های KeyPress و KeyDown زمانی که کلیدی فشار داده شود، فراخوانی می‌شوند. هنگام که کلید فشار داده شده رها شود، روال KeyUp فراخوانی خواهد شد. KeyPress متفاوت با KeyDown است. KeyPress نمی‌تواند تشخیص دهد که کدام کلید مانند Shift، Ctrl و Alt و غیره فشار داده شده است.

بسته به این که کدام کلید فشار داده شده باشد، هر سه روال ممکن است فراخوانی شوند. جدول ۹ــ۱ شامل لیستی از ثابت‌هاست که مربوط به رویدادهای کلید هستند.

برنامهٔ ۴ــ۱ نحوهٔ استفاده از رویدادهای کلید را نشان می‌دهد. هنگامی که کلیدی فشار داده شود، keyDown و KeyPress فراخوانی می‌شوند. از keyDown برای چاپ کلید فشار داده شده بر روی فرم و از KeyPress برای نمایش کلید فشار داده شده در نوار عنوان فرم استفاده می‌شود. همچنین KeyDown وضعیت کلیدهای Ctrl ، Shift و Alt را برای تغییر رنگ رویهٔ فرم بررسی می کند. KeyUp رنگ رویهٔ فرم را با رنگ سیاه تنظیم می کند.

جدول ۹ــ۱ــ ثابت‌های مربوط به رویدادهای کلید

ثابت‌ها	مقدار اسکی (ASCII)	توضیحات
vbKeyA–vbKeyZ	65–90	کلیدهای A تا Z
vbKeyNumpad0–vbKeyNumpad9	95–105	اعداد 0 تا 9 بخش عددی صفحه کلید
vbKey0–vbKey9	48–57	کلیدهای عددی 0 تا 9
vbKeyF1–vbKeyF16	112–127	کلیدهای تابعی F1 الی F16
vbKeyDecimal	110	کلید نقطه اعشار (کلید نقطه)
vbKeyBack	8	کلید Backspace
vbKeyTab	9	کلید Tab
vbKeyReturn	13	کلید Enter
vbKeyShift	16	کلید Shift
vbKeyControl	17	کلید Control
vbKeyCapital	20	کلید Caps Lock
vbKeyEscape	27	کلید Esc
vbKeySpace	32	Space bar
vbKeyInsert	45	کلید Insert
vbKeyDelete	46	کلید Delete

برنامهٔ ۴ــ۱ استفاده از رویدادهای کلید

```
1. ‘Demonstrating KeyDown, KeyUp, and, KeyPress
2. Option Explicit          ‘General declaration
3. Dim mTitleString As String    ‘General declaration
4. Private Sub Form_load ()
5.          ‘Store caption value for use in Keypress
6.          mTitleString = Caption & Space$ (5)
7.  End Sub
8. Private Sub Form_KeyDown (KeyCode As Integer, Shift As Integer)
9.          ‘Determine which, if any, of the Shift, Ctrl,
10.         ‘or Alt keys is pressed
11.         Select Case Shift
12.         Case vbShiftMask       ‘Shift
13.         ForeColor = vbYellow
14.         Case vbAltMask         ‘Alt
15.         Fore Color = vbRed
16.         Case vbCtrlMask          ‘Ctrl
17.         ForeColor = vbGreen
18.          Case vbShiftMask + vbAltNask ‘Shift + Alt
19.         ForeColor = vbBlue
20.         Case vbShiftMask + vbCtrlMask          ‘Shift + Ctrl
21.          ForeColor = vbMagenta
22.          Case vbAltMasl + vbCtrlMask       ‘Alt + Ctrl
23.         ForeColor = vbCyan
24.         Case vbAltMask + vbCtrlMask + vbShiftMask ‘All three
25.          Cls
26.         End Select
```

```
27.         'Test for letter key
28.      If KeyCode > = vbKeyA And KeyCode < = vbKeyZ Or KeyCode =
         vbKeySpace Then
29.          Print Chr (KeyCode);    'Print the character
30.      ElseIf KeyCode = vbKeyreturn Then    'Return key
31.          Print    'Print on next line
32.       End If
33. End Sub
34. Private Sub Form_KeyPress (KeyAscii as Integer)
35.              'Update title to display the key Pressed
36.      Caption = mTitleString & "("& Chr (key Ascii) & ")"
37. End Sub
38. Private Sub Form_KeyUp(KeyCode as Integer, Shift As Integer)
39. When key is released, change ForeColor to black
40.   ForeColor = vbBlack
41. End Sub
```

روال KeyDown با استفاده از متغیر Shift، وضعیت کلیدهای Shift، Ctrl و Alt را در زمان رخ دادن رویداد KeyDown مشخص می‌کند. اگر هریک از این کلیدها فشار داده شوند، این رویداد اتفاق می‌افتد و رنگ رویه تغییر پیدا می‌کند. پارامتر KeyCode شامل یک مقدار اسکی است که نشان‌دهندهٔ یک کلید فشار داده شده است. برای تشخیص حرف، از KeyCode استفاده شده است. KeyCode بین حروف کوچک و بزرگ تفاوت قایل می‌شود. از تابع Chr برای تبدیل کدهای اسکی به نویسه استفاده شده است (مثلاً ۶۵ به "A" تبدیل می‌شود). در روال keyPress هم از Chr برای تبدیل مقدار اسکی به رشته استفاده شده است. متغیر KeyAscii بین حروف کوچک و بزرگ تفاوت قایل است. روال KeyPress شامل پارامتر Shift نیست. هرگاه KeyUp مربوط به فرم رخ دهد، دوباره رنگ رویه به رنگ سیاه تغییر می‌یابد. با فشار کلیدهای Shift، Ctrl و Alt نیز رنگ رویه تغییر می‌کند ولی با فشار همزمان هر سه کلید، فرم پاک خواهد شد (شکل ۵ــ۱).

نکته

فقط نویسه‌های A تا Z و فضای خالی روی فرم چاپ می‌شوند و علایم فقط در نوار عنوان فرم مشاهده می‌شوند.

شکل ۵ــ۱

۱ــ۳ــ۱ــ مشخصهٔ KeyPreview : اگر فرمی شامل حداقل یک کنترل قابل مشاهده (در این‌جا کنترلی مد نظر است که مشخصهٔ Visible آن با True تنظیم شده باشد) یا یک کنترل فعال باشد، آن فرم نمی‌تواند به رویدادهای کلید واکنش نشان دهد. برای تغییر این وضعیت، باید مشخصهٔ KeyPreview فرم با True مقداردهی شود تا رویدادهای کلید قبل از آن‌که به وسیلهٔ کنترل‌ها دریافت شوند، به وسیلهٔ فرم دریافت شوند.

معمولاً از KeyPreview هنگامی استفاده می‌شود که چندین کنترل نیاز به یک واکنش در برابر رویدادهای کلید داشته باشند.

برنامهٔ ۵ــ۱ نحوهٔ استفاده از KeyPreview را نشان داده است. برنامه با استفاده از این مشخصه مطمئن می‌شود که فقط رقم‌ها وارد کادر متن می‌شوند.

در روال Form_Load مشخصهٔ KeyPreview با True مقداردهی می‌شود. با این عمل، فرم می‌تواند جلوی رویدادهای کلید را قبل از آنکه به‌صورت خودکار، به روال KeyPress کادر متن یا کادرهای متن انتقال پیدا کنند، بگیرد. هنگامی که کاربر دادهٔ مورد نظرخود را از طریق کادر متن وارد می‌کند، روال KeyPress فرم فراخوانی می‌شود و عمل بررسی بر روی نویسه‌های رقم انجام می‌گیرد (شکل ۶ــ۱).

شکل ۶ــ۱

برنامهٔ ۵ــ۱ استفاده از مشخصهٔ KeyPreview

```
1.  'Demontrating  the KeyPreview property
2.  Option Explicit        'General declaration
3.  Private Sub form_Load ()
4.      'Allow Form to get key events first
5.      Keypreview = True
6.  End Sub
7.  Private Sub Form_KeyPress (KeyAscii As Integer)
8.      'Only allow numeric keys
9.      If KeyAscii>=vbKey0 And KeyAscii<=vbKey9 Then
10.         txtInput.Text = txtInput.Text & Chr (key Ascii)
11.     End If
12.     Ens Sub
13.     Private Sub txtInput_KeyPress (KeyAscii As Integer)
14.         Key Ascii = 0        'Disable event handling
15.     End Sub
16.     Private Sub txtInput_KeyUp (KeyCode As Integer, Shift As Integer)
17.         Selcet Case Int (Rnd() * 3)
18.         Case 0
19.         txtInput.BackColor = vbYellow
20.         Case 1
21.         txtInput.BackColor = vbCyan
22.         Case 2
23.         txtInput.BackColor =  vbRed
24.         End Select
25.  End Sub
```

برای جلوگیری از فراخوانی‌های اضافی روال KeyPress کادر متن، مکانیزم دستیابی به این

روال را به‌صورت KeyAscii=0 غیرفعال کرده‌ایم. هرگاه KeyAscii=0 شود، مجوز عبور کلید فشار داده شده سلب می‌شود ولی در مورد KeyCode چنین نیست. روال KeyUp به‌صورت تصادفی رنگ زمینهٔ متن را تغییر می‌دهد. زمانی که KeyUp غیرفعال نشده باشد، در هر بار رهاکردن کلید، این روال فراخوانی می‌شود.

هرگاه یکی از کلیدهای زیر فشار داده شود، رویداد KeyPress فراخوانی خواهد شد :

- حروف بزرگ یا کوچک
- اعداد
- علایم نقطه‌گذاری
- Backspace و Tab، Enter

رویداد Keypress، اکثر نویسه‌های اسکی را تشخیص می‌دهد (ولی نمی‌تواند نویسه‌های ۰ تا ۳۱ جدول اسکی را تشخیص دهد). رویداد Keypress با فشاردادن کلید فراخوانی می‌شود و اگر کاربر دستش را روی کلید نگه دارد، با تکرار نویسه‌ها این رویداد هم تولید خواهد شد.

همانطورکه می‌دانید، هر رویداد به یک شیء وابسته است. رویداد KeyPress هم به کنترلی که فوکوس را در اختیار دارد، وابسته است. اگر هیچ کنترلی فوکوس نداشته باشد، این رویداد به فرم خواهد رسید.

رویداد KeyPress دارای یک آرگومان از نوع Integer است. رویداد Key Press برای یک کادر متن به صورت زیر است :

```
Private Sub Text_KeyPress (KeyAscii As Integer)
    ‘Code goes here test and respond to keystroke
End Sub
```

آرگومان KeyAscii معادل کد اسکی کلید زده شده خواهد بود و با یک دستور If یا Case Select می‌توان آن را بررسی کرد.

یکی از مهمترین کارهایی که می‌توان با KeyPress انجام داد، تغییردادن کلید زده شده به وسیلهٔ کاربر است، چون کلید زده شده قبل از اینکه به کنترل‌ها برسد از این رویداد عبور خواهد کرد. به‌عنوان مثال، به یاد دارید که وقتی فوکوس در اختیار یک کادر متن است، هر کلیدی که کاربر بزند، بلافاصله در کادر متن ظاهر خواهد شد. اما با رویداد KeyPress می‌توانید روالی بنویسید که کلید زده شده را عوض کند :

```
Private Sub txtTrylt_KeyPress (KeyAscii As Integer)
   ‘ Change any uppercase A to an uppercase B
If KeyAscii = 65 Then    ‘65 is ASCII for A
    Key Ascii = 66        ‘66 is ASCII for B
End If
End Sub
```

این کد سبب می‌شود که هرگاه کاربر A را بزند، کادر متن B را نشان دهد! (حروف دیگر بدون هیچ تغییری نشان داده خواهند شد). لازم به ذکر است که نیازی نیست کد اسکی کلیدها را حفظ کنید چون ویژوال بیسیک برای تمام آن‌ها ثابت‌های نام‌دار تعریف کرده است. به‌عنوان مثال، برای Backspace ثابت vbKeyBack و برای کلید Enter ثابت vbKeyReturn و برای کلید Tab ثابت vbKeyTab (و الی آخر) تعریف شده‌اند. که در جدول ۹ــ۱ آن‌ها را مشاهده کردید.

رویداد KeyDown (که با پایین‌رفتن کلید، تحریک می‌شود) مانند KeyPress است ولی جزییات بیشتری در اختیار برنامه‌نویس قرار می‌دهد. به‌عنوان مثال، رویداد KeyPress برای کلیدهای T یا t کدهای متفاوتی برمی‌گرداند، ولی رویداد KeyDowm برای این دو کلید کد یکسانی برمی‌گرداند و آرگومانی دارد که حالت کلید Shift، Ctrl و Alt را نشان می‌دهد. کار با رویداد KeyPress برای کنترل کلیدهای اسکی ساده‌تر است. بنابراین تا حد امکان از آن استفاده کنید. شکل کلی روال رویداد KeyDown چنین است :

```
Private Sub txtTryIt_KeyDown (KeyCode As Integer, Shift As Integer)
        ‘Keyboard code handler goes here
End Sub
```

آرگومان KeyCode، کد کلیدزده شده و آرگومان Shift، وضعیت کلیدهای Shift، Ctrl و Alt را در خود خواهند داشت. آرگومان KeyCode همیشه معادل حروف بزرگ کلید زده شده است. اگر دقت نکنید، این وضعیت می‌تواند موجب سردرگمی شما شود.

مزیت رویداد KeyDown (علیرغم مشکل فوق) این است که با آن می‌توان تقریباً تمام کلیدهای صفحه کلید را تشخیص داد : کلیدهایی مانند Home، End، کلیدهای فلش‌دار و غیره. آرگومان Shift نوع کلید Shift، Ctrl یا Alt را مشخص خواهد کرد.

وقتی کاربر کلید فشار داده شده را رها کند، رویداد KeyUp فعال خواهد شد. در این رویداد

هم می‌توان وضعیت کلیدهای ترکیبی را بررسی کرد. توجه دارید که هر رویداد KeyPress ترکیبی از رویدادهای KeyDown و KeyUp است.

۲ــ۳ــ۱ــ ارسال ضربات کلید به برنامه : دستور SendKeys می‌تواند ضربات کلید را به برنامه بفرستد. از نظر برنامه، هیچ تفاوتی بین دستور SendKeys و ضربات واقعی صفحه کلید وجود ندارد. شکل کلی این دستور چنین است :

SendKeys strKeystrokes [,blnWait]

رشتهٔ strKeystrokes، همان نویسه‌هایی است که باید ارسال شوند. اگر آرگومان منطقی blnWait (که معمولاً ذکر نمی‌شود) False باشد (مقدار پیش‌فرض)، روال اجراکنندهٔ دستور SendKeys بلافاصله بعد از ارسال نویسه‌ها مجدداً کنترل برنامه را در دست می‌گیرد. اما اگر این آرگومان True باشد، سیستم تا پردازش کامل نویسه‌ها، کنترل را به روال اجراکنندهٔ دستور SendKeys بازپس نخواهد داد.

برای ارسال نویسه‌های خاص (^، +، ~، (،)، %) باید آن‌ها را داخل یک زوج آکولاد {} قرار دهید. به‌عنوان مثال، اگر می‌خواهید تایپ ۶+۷ را برای برنامه شبیه سازی کنید، باید از دستور SendKeys زیر استفاده کنید :

```
SendKeys "7{+}6"
```

یا برای ارسال کلید Home به برنامه، باید چنین بنویسید :

```
SendKeys "{Home}"
```

تنها کلیدی که نمی‌توان با SendKeys آن را به برنامه‌ها فرستاد، کلید Print Screen است.

جدول ۱۰ــ۱ کلیدهای ویژه را نشان می‌دهد.

جدول ۱۰ــ۱

کلیدهای ویژه	عبارت معادل
BACKSPACE	{BACKSPACE}, {BS}, or {BKSP}
BREAK	{BREAK}
CAPS LOCK	{CAPSLOCK}
DEL or DELETE	{DELETE} or {DEL}

Down Arrow	{DOWN}
END	{END}
ENTER	{ENTER} or ~
ESC	{ESC}
HELP	{HELP}
HOME	{HOME}
INS or INSERT	{INSERT}or {INS}
LEFT ARROW	{LEFT}
NUM LOCK	{NUMLOCK}
PAGE DOWN	{PGDN}
PAGE UP	{PGUP}
PRINT SCREEN	{PRTSC}
RIGHT ARROW	{RIGHT}
SCROLL LOCK	{SCROLLLOCK}
TAB	{TAB}
UP ARROW	{UP}
F1	{F1}
F2	{F2}
F3	{F3}
F4	{F4}
F5	{F5}
F6	{F6}
F7	{F7}
F8	{F8}

F9	{F9}
F10	{F10}
F11	{F11}
F12	{F12}
F13	{F13}
F14	{F14}
F15	{F15}
F16	{F16}

برای استفاده از کلیدهای Shift، Ctrl و Alt باید از جدول ۱۱ــ۱ استفاده کرد :

جدول ۱۱ــ۱

Key	Code
SHIFT	+
CTRL	^
ALT	%

به‌عنوان مثال، "(EC)+" یعنی فشاردادن کلید Shift به همراه کلیدهای E و C و "EC+" به معنی فشاردادن کلید Shift به همراه کلید E و پس از رهاکردن، فشاردادن کلید C است.

برای تکرار یک کلید، لازم است از مدل {key number} استفاده کنیم که در آن Key همان کلید دلخواه و number تعداد دفعات تکرار است. توجه داشته باشید که باید بین Key و number فاصله وجود داشته باشد.

به‌عنوان مثال، {A 5} یعنی فرستادن ۵ بار نویسهٔ A ، {Right 10} یعنی فرستادن کلید جهت‌دار راست به تعداد ۱۰ مرتبه.

دریافت داده‌ها از ورودی و عملیات شرطی

در این مثال با نحوهٔ استفاده از کنترل‌های کادر علامت، دکمهٔ انتخاب و فریم و متدهای Show

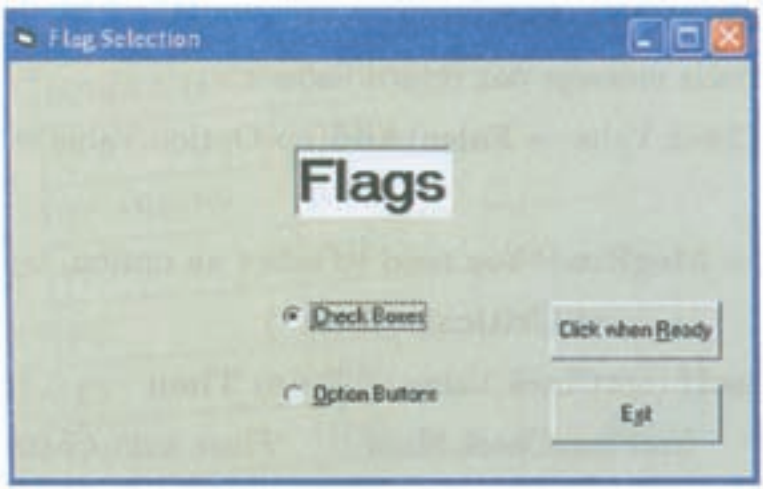
و Hide در یک برنامهٔ چندفرمی آشنا خواهید شد. گرفتن اطلاعات از کاربر و کنترل پاسخ‌های وی هم از اهداف آموزشی این مثال است. این مثال به کدنویسی زیادی احتیاج دارد. در این پروژه :

- چند کادر علامت وجود دارد که کاربر با انتخاب آن‌ها می‌تواند تا پرچم کشورهای مربوطه را مشاهده کند.
- کاربر می‌تواند همان وظیفهٔ فوق را با دکمه‌های انتخاب انجام دهد.
- اگر کاربر به یک درخواست، پاسخ صحیح ندهد، با یک کادر پیام به وی اخطار داده می‌شود.
- کاربر می‌تواند نحوهٔ نمایش (بزرگ و کوچک) پرچم‌ها را با یک سری دکمهٔ انتخاب تغییر دهد.

این فرم از پرونده‌های گرافیکی Visual Basic 6 استفاده می‌کند و اگر هنگام نصب Visual Basic 6 آن‌ها را در هارددیسک خود کپی نکرده اید، باید مسیر پرونده را با محل اصلی آن‌ها (CD نصب ویژوال بیسیک) تنظیم کنید.

فرم اولیهٔ برنامه

شکل ۷ــ۱ اولین فرم برنامه را نشان می‌دهد. جدول ۱۲ــ۱ مشخصه‌های کنترل‌های فرم اولیهٔ برنامه را ارایه می‌دهد.



شکل ۷ــ۱

جدول ۱۲ــ۱ــ مشخصه‌های کنترل‌های فرم اولیهٔ برنامه

مشخصه	مقدار
Form Name	FrmSelect
Form caption	Flag Selection
Label Name	lblFlags
Label BorderStyle	1–Fixed Single
Label caption	Flage
Label Font	MS Sans Serif
Label Font Size	24
Label Font Style	Bold
Option Button #1 Name	optCheck
Option Button #1 caption	& Check Boxes
Option Button #2 Name	optOption
Option Button #2 caption	& Option Buttons
Command buttom #1 Name	cmdSelect
Command buttom #1 Caption	Click when & Ready
Command buttom #2 Name	cmdExit
Command buttom #2 Caption	E&xit

کد این فرم، به‌صورت زیر است :

```
1. Private Sub cmdSelect_Click ()
2.     Dim strMsg As String
3.     'Holds message box return value
4. If ((optCheck.Value = False) And (optOption.Value = False)) Then
5. strMsg = MsgBox ("You need to select an option, try again", _
```

```
6.                  vbCritical, "Error")
7.        ElseIf (optCheck.Value = True) Then
8.             frmFlageCheck.Show     'Flage with Check boxes
9.        Else
10.       frmFlagesOpt.Show          'Flage with Option buttons
11.  End If
12.  End Sub
13.  Private Sub Form_Load ()
14.       'Clear each of the option buttons
15.       optCheck.Value = False
16.       optOption.Value = False
17.  End Sub
18.  Private Sub  cmdExit_Click ()
19.       'Stop the program
20.       End
21.  End Sub
```

تحلیل کد فرم اولیه : خطوط ۱۴ تا ۱۸ برنامهٔ فوق، اعمالی هستند که هنگام اجرای برنامه و بارشدن اولین فرم آن انجام می‌شوند (توجه داشته باشید که باید در کادر محاوره‌ای Project | Properties فرم FrmSelect به‌عنوان فرم Startup برنامه تنظیم شده باشد). این روال خاصیت Value هر دو دکمهٔ انتخاب این فرم را False می‌کند تا کاربر مجبور به انتخاب یکی از آن‌ها شود. به عبارت دیگر، در این فرم گزینهٔ پیش‌فرض وجود ندارد.

اگر کاربر بدون انتخاب یکی از این گزینه‌ها، دکمهٔ فرمان cmdSelect را کلیک کند، دستور If خط ۴، متوجه این وضعیت شده و به کاربر پیام می‌دهد که بایستی یکی از گزینه‌ها را انتخاب کند.

اما اگر به درستی عمل کرده باشد و یکی از گزینه‌ها را انتخاب و سپس cmdSelect را کلیک کرده باشد، خط‌های ۷ تا ۹ تعیین می‌کنند که کدام گزینه انتخاب شده است، چون فرمی که باید نمایش داده شود به گزینهٔ انتخاب شده در فرم اولیهٔ برنامه بستگی دارد. برای نمایش فرم‌ها از متد Show استفاده می‌شود. این متد فرمانی است که فقط روی شیء فرم عمل می‌کند و نحوهٔ استفاده از آن مانند سایر متدهاست.

ویژوال بیسیک با دیدن متد Show، فرم مشخص شده را در حافظه بارکرده و نمایش می‌دهد.

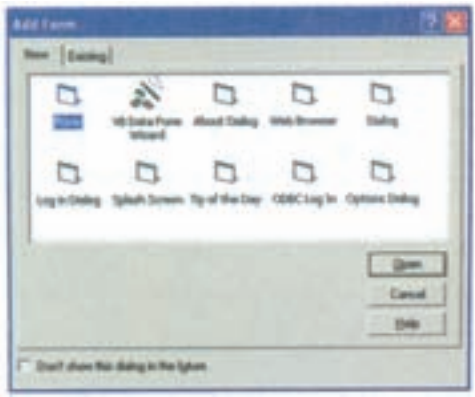
فرم کادر علامت : در شکل ۸ــ۱ فرم دوم برنامه یعنی فرمی که با استفاده از کادر علامت، پرچم‌ها رانمایش می‌دهد، نشان داده شده است.

شکل ۸ــ۱

در این فرم، شش کادر علامت وجود دارد که هریک پرچم کشوری را نمایش می‌دهند. اما برای ایجاد یک فرم جدید در پروژه :

۱ــ داخل پنجرهٔ پروژه کلیک راست کنید.

۲ــ از منوی ظاهر شده، گزینهٔ Add|Form را انتخاب کنید؛ کادر محاوره‌ای ظاهر می‌شود که در آن می‌توانید یک فرم جدید (یا یک فرم موجود) را به پروژه اضافه کنید (شکل ۹ــ۱).



شکل ۹ــ۱

۳ــ روی نشانهٔ Form دو بار کلیک کنید تا فرم جدید به برنامه اضافه شود.

در جدول ۱۳ــ۱ مشخصه‌های کنترل‌های روی این فرم را مشاهده می‌کنید. به یاد داشته باشید که مسیر پرونده مشخصهٔ Picture کنترل‌های تصویر را باید مطابق آنچه در رایانه‌تان وجود دارد، مقداردهی کنید.

جدول ۱۳ــ۱ــ مشخصه‌های کنترل‌های فرم کادر علامت

مشخصه	مقدار
Check box #1 Name	chkBrazil
Check box #1 Caption	& Brazil
Check box #2 Name	chkItaly
Check box #2 Caption	& Italy
Check box #3 Name	chkSpain
Check box #3 Caption	& Spain
Check box #4 Name	chkMexico
Check box #4 Caption	& Mexico
Check box #5 Name	chkportegal
Check box #5 Caption	& portegal
Check box #6 Name	chkNORW
Check box #6 Caption	& NORW
Image #1 Name	imgBrazil
Image #1 Picture	\Progran Files\Microsoft Visual Studio\ common\Graphics\Icons\Flags\FlagBrzl
Image #1Visible	False
Image # 2 Name	ImgItaly
Image #2 Picture	\Program Files\Microsoft Visual Studio\ common\Graphics\Icons\Flags\FlagItaly

Image #2Visible	False
Image #3 Name	ImgSpain
Image #3 Picture	\Progran Files\Microsoft Visual Studio\ common\Graphics\Icons\Flags\FlagPain
Image #3Visible	False
Image #4 Name	imgMexico
Image #4 Picture	\Program Files\Microsoft Visual Studio\ common\Graphics\Icons\Flags\Flagmex
Image #4 Visible	False
Image #5 Name	ImgPortegal
Image #5 Picture	\Program Files\Microsoft Visual Studio\ common\Graphics\Icons\Flags\FlagPort
Image #5 Visible	False
Image #6 Name	ImgNORW
Image #6 Picture	\Program Files\Microsoft Visual Studio\ common\Graphics\Icons\Flags\'FlagsNorw
Image #6 Visible	False
Comm and buttonName	cmdReturn
command button Caption	& Return to selection

حال روی فرم frmFlagCheck دو بار کلیک کرده و کد برنامهٔ ۶ــ۱ را در پنجرهٔ کد این فرم بنویسید. چون با کلیک‌کردن هر کادر علامت، باید تصویر متناظر با آن به‌طور متناوب ظاهر و پنهان شود. برای رویداد Click هر کادر علامت، یک روال اختصاصی نوشته شده است.

برنامهٔ ۶ــ۱ کد فرم کادر علامت

```
1. Private Sub chkBrazil_Click()
2. 'Displays the flag if checked
```

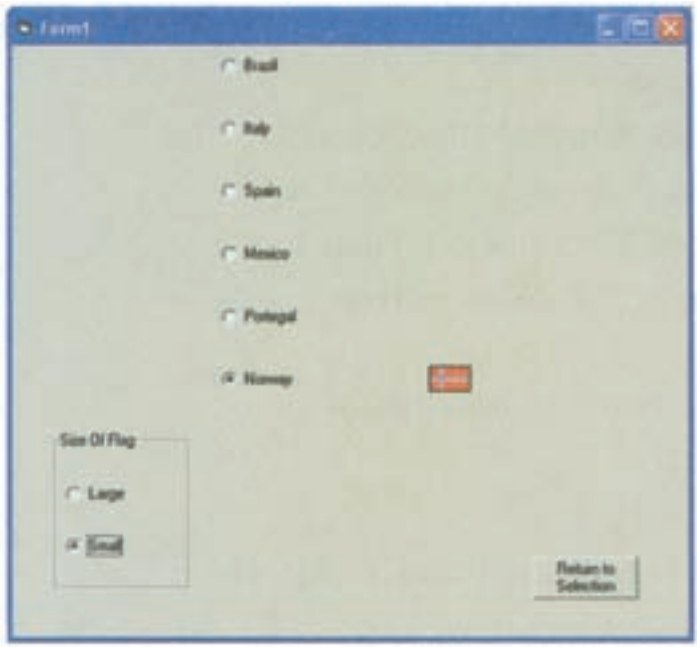
```
3. If chkBrazil.Value = 1 Then
4.         imgBrazil.Visible = True
5. Else
6.         imgBrazil.Visible = False
7. End If
8. End Sub
9. Private Sub chkItaly_Click()
10.        'Displays the flag if checked
11.         If chkItaly.Value = 1 Then
12.            imgItaly.Visible = True
13.         Else
14.            imgItaly.Visible = False
15.  End If
16.  End Sub
17.  Private Sub chkportegal_Click ()
18.        'Displays the flag if checked
19.      If chkportegal.Value = 1 Then
20.         imgportegal.Visible = True
21.     Else
22.        imgportegal.Visible = False
23. End If
24. End Sub
25.  Private Sub chkMexico_Click()
26. 'Displays the flag if checked
27. If chkMexico.Value = 1 Then
28.         imgMexico.Visible = True
29.   Else
```

```
30.       imgMexico. Visible = False
31.  End If
32.  End Sub
33.  Private Sub chkSpain_Click ()
34.  'Displays the flag if checked
35.  If chkSpain.Value = 1 Then
36.       imgSpain.Visible = True
37.  Else
38.       imgSpain.Visible = False
39.  End If
40.  End Sub
41.  Private Sub chkNORW_Click ()
42.  'Displays the flag if checked
43.  If chkNORW.Value = 1 Then
44.       imgNORW.Visible = True
45.  Else
46.       imgNORW.Visible = False
47.       End If
48.       End Sub
49.       Private Sub cmdReturn_Click()
50.          frmFlageCheck.Hide
51.          FrmSelect.Show
52.       End Sub
```

تحلیل کد فرم کادر علامت : در برنامهٔ ۶ــ۱ شش روال رویداد کاملاً مشابه وجود دارد و کار آن‌ها فقط مقداردهی کردن مشخصهٔ Visible کنترل تصویر متناظر با هر کادر علامت است. به دستور If دقت کنید : اگر کاربر روی کادر علامتی که فعال است کلیک کند، قسمت Else اجرا شده وسبب ناپدیدشدن تصویر خواهد شد. هر روال رویداد برای اطلاع از وضعیت فعلی کادر علامت به

مقدار مشخصهٔ Value آن توجه می‌کند : اگر این خاصیت ۱ باشد، تصویر ظاهر و اگر ۰ باشد، تصویر پنهان می‌شود. علاوه‌بر روال‌های کادر علامت، روال cmdReturn_Click نیز در این برنامه مشاهده می‌شود. این روال، فرم کادر علامت را از روی صفحه ناپدید کرده و مجدداً فرم frmSelect را نمایش می‌دهد. وظیفهٔ متد Hide عکس متد Show است.

فرم دکمهٔ انتخاب : در شکل ۱۰ــ۱ فرم سوم پروژه، یعنی فرمی که با دکمه‌های انتخاب، پرچم‌ها را نمایش می‌دهد، نشان داده شده است. این فرم یک امکان دیگر هم در اختیار کاربران قرار می‌دهد : پرچم‌های بزرگ یا پرچم‌های کوچک.



شکل ۱۰ــ۱

این فرم هم مانند فرم کادر علامت، دارای شش دکمهٔ انتخاب (برای هریک از کشورها) است. علاوه‌برآن، این فرم دارای یک فریم با دو دکمهٔ انتخاب برای تعیین اندازهٔ نمایش پرچم‌هاست.

فرم سوم را هم مانند قبل بسازید و مشخصه‌های کنترل‌های آن را مانند جدول ۱۴ــ۱ مقداردهی کنید.

جدول ۱۴ــ۱ــ خواص کنترل‌های فرم دکمهٔ انتخاب

مشخصه	مقدار
Form Name	frmFlagsOpt
Form Caption	Flags
Option button # 1 Name	optBrazil
Option button # 1 Caption	& Brazil
Option button # 1 Value	True
Option button # 2 Name	optItaly
Option button # 2 Caption	& Italy
Option button # 3 Name	optSpain
Option button # 3 Caption	& Spain
Option button # 4Name	optMexico
Option button # 4 Caption	& Mexico
Option button # 5 Name	optportegal
Option button # 5 Caption	& portegal
Option button # 6 Name	optNORW
Option button # 6 Caption	& NORW
Frame Name	fraSize
Frame Caption	Size the flag
Frame Option # 1 Name	optLarge
Frame Option # 1 Caption	& Large
Frame Option # 2 Name	optSmall
Frame Option # 2 Caption	Sma&ll
Image # 1 Name	imgBrazil

Image # 1 Picture	\Program\Microsoft Visual Studio\Common\ Graphics\Icons\Flags\FlgBrzil
Image # 1 Stretch	True
Image # 1 Visible	True
Image # 2 Name	ImgItaly
Image # 2 Picture	\Program\Microsoft Visual Studio\Common\ Graphics\Icons\Flags\Flgitaly
Image # 2 Stretch	True
Image # 2 Visible	False
Image # 3 Name	ImgSpain
Image # 3 Picture	\Program\Microsoft Visual Studio\Common\ Graphics\Icons\Flags\FlgSpain
Image # 3 Stretch	True
Image # 3 Visible	False
Image # 4 Name	ImgMexico
Image # 4 Picture	\Program\Microsoft Visual Studio\Common\ Graphics\Icons\Flags\Flgmex
Image # 4 Stretch	True
Image # 4 Visible	False
Image # 5 Name	Imgportegal
Image # 5 Picture	\Program\Microsoft Visual Studio\Common\ Graphics\Icons\Flags\Flgport
Image # 5 Stretch	True

Image # 5 Visible	False
Image # 6 Name	ImgNORW
Image # 6 Picture	\Program\Microsoft Visual Studio\Common\ Graphics\Icons\Flags\FlgNorw
Image # 6 Stretch	True
Image # 6 Top	4080
Image # 6 Visible	False
Command button Name	Cmdreturn
Command button Caption	& Return to selection

کد فرم frmFlagsOpt را در برنامهٔ ۷ــ۱ مشاهده می‌کنید. این کد کمی طولانی است! ولی بیشترین بخش آن روال‌های تکراری کلیک‌شدن روی دکمه‌های انتخاب است.

برنامهٔ ۷ــ۱ کد فرم دکمهٔ انتخاب

```
1. Private Sub optBrazil_Click()
2.       'Displays the flag if checked
3.    If optSmall.Value = True Then
4.       imgBrazil.Height = 480
5.       imgBrazil.Width = 480
6.    Else    'Large image
7.       imgBrazil.Height = 2800
8.       imgBrazil.Width = 2800
9.    End If
10.   imgBrazil.Visible = True
11.   'Turn off display of all other flags
12.   imgItaly.Visible = False
13.   imgSpain.Visible = False
```

```
14.     imgMexico.Visible = False
15.     imgportegal.Visible = False
16.     imgNORW.Visible = False
17. End Sub
18. Private Sub optItaly_Click()
19.     'Displays the flag if checked
20.     If optSmall.Value = True Then
21.         imgItaly.Height = 480
22.         imgItaly.Width = 480
23.     Else     'Large image
24.         imgItaly.Height = 2800
25.         imgItaly.Width = 2800
26.     End If
27.     imgItaly.Visible = True
28.     'Turn off display of all other flags
29.     imgBrazil.Visible = False
30.     imgSpain.Visible = False
31.     imgMexico.Visible = False
32.     imgportegal.Visible = False
33.     imgNORW.Visible = False
34. End Sub
35. Private Sub opts Pain_Click()
36.     'Display the flag if checked
37.     If optSmall.Value = True Then
38.         imgSpain.Height = 480
39.         imgSpain.Width = 480
40.     Else     'Large image
41.         imgSpain.Height = 2800
```

```
42.          imgSpain.Width = 2800
43.     End If
44.      imgSpain.Visible = True
45.     'Turn off display of all other flags
46.      imgItaly.Visible = False
47.     imgBrazil.Visible = False
48.      imgMexico.Visible = False
49.     imgportegal.Visible = False
50.      imgNORW.Visible = False
51. End Sub
52. Private Sub opMexico_Click()
53.     'Displays the flag if checked
54.     If optSmall.Value = True Then
55.          imgMexico.Height = 480
56.          imgMexico.Width = 480
57.     Else     'Large image
58.          imgMexico.Height = 2800
59.          imgMexico.Width = 2800
60.     End If
61.     imgMexico.Visible = True
62.     'Turn off display of all other flags
63.     imgItaly.Visible = False
64.      imgSpain.Visible = False
65.      imgBrazil.Visible = False
66.     imgportegal.Visible = False
67.     imgNORW.Visible = False
68. End Sub
69. Private Sub optportegal_Click()
```

```
70.      'Displays the flag if checked
71.     If optSmall.Value = True Then
72.           imgportegal.Height = 480
73.           imgportegal.Width = 480
74.     Else    'Large image
75.           imgportegal.Height = 2800
76.           imgportegal.Width = 2800
77.     End If
78.    imgportegal.Visible = True
79.     'Turn off display of all other flags
80.     imgItaly.Visible = False
81.     imgSpain.Visible = False
82.     imgMexico.Visible = False
83.     imgBrazil.Visible = False
84.     imgNORW.Visible = False
85. End Sub
86. Private Sub optNorw_Click()
87.    'Displays the flag if checked
88.     If optSmall.Value = True Then
89.           imgpNORW.Height = 480
90.           imgNORW.Width = 480
91.    Else    'Large image
92.          imgNORW.Height = 2800
93.          imgNORW.Width = 2800
94.    End If
95.     imgNORW.Visible = True
96.     'Turn off display of all other flags
97.     imgItaly.Visible = False
```

```
98.     imgSpain.Visible = False
99.     imgMexico.Visible = False
100.    imgportegal.Visible = False
101.    imgBrazil.Visible = False
102. End Sub
103. Private Sub cmdReturn_Click()
104.    'Return to the selection form
105.    frmFlagsOpt. Hide
106.    frmSelect. Show
107. End Sub
108. Private Sub optSmall_Click()
109.        'Hide all flags shown
110.        'Subsequent flags will be small
111.        imgBrazil.Visible = False
112.        imgItaly.Visible = False
113.        imgSpain.Visible = False
114.        imgMexico.Visible = False
115.        imgportegal.Visible = False
116.        imgNORW.Visible = False
117.        'Reset option buttons
118.        optBrazil.Value = False
119.        optItaly.Value = False
120.        optSpain.Value = False
121.        optMexico.Value = False
122.        optportegal.Value = False
123.        optNORW.Value = False
124. End Sub
125. Private Sub optLarge_Click()
```

```
126.        'Hide all flags shown
127.        'Subsequent flags will be large
128.        imgBrazil.Visible = False
129.        imgItaly.Visible = False
130.        imgSpain.Visible = False
131.        imgMexico.Visible = False
132.        imgportegal.Visible = False
133.        imgNorw.Visible = False
134.        'Reset option buttone
135.        optBrazil.Value = False
136.        optItaly.Value = False
137.        optSpain.Value = False
138.        optMexico.Value = False
139.        optportegal.Value = False
140.        optNORW.Value = False
141. End Sub
```

تحلیل کد فرم دکمهٔ انتخاب : قسمت اعظم کد تکراری است. به خط ۱ تا ۱۷ آن دقت کنید. در اینجا ابتدا روال بررسی می‌کند که آیا تصویر باید بزرگ باشد یا کوچک و برنامه با مقداردهی Height و Width کنترل تصویر، اندازهٔ مناسب را انتخاب می‌کند. ادامهٔ روال به ظاهرکردن تصویر مربوطه (خط ۱۰) و ناپدیدکردن سایر تصویرها (خط‌های ۱۲ تا ۱۶) اختصاص دارد. سایر روال‌ها دقیقاً مشابه این روال هستند.

نکته

در ادامه کتاب خواهید آموخت که با نوشتن توابع می‌توان این کد را ساده‌تر کرد.

خط‌های ۱۰۳ تا ۱۰۷ روال پنهان‌کردن این فرم و ظاهرکردن دوباره فرم اصلی برنامه است. هنگام کلیک‌کردن دکمه‌های OptSmall و OptLarge، برنامه (برای راحتی کار خود) تمام گزینه‌ها را False و تمام پرچم‌ها را ناپدید می‌کند.

خلاصه فصل

در زبان برنامه نویسی ویژوال بیسیک، می‌توان به کمک کنترل‌های Image و PictureBox از تصویرهای با قالب‌های مختلف استفاده کرد. با استفاده از تابع LoadPicture می‌توان تصویری را در مشخصهٔ Picture این کنترل‌ها قرار داد.

زمانی که برنامه‌های کاربردی ایجاد می‌کنید، نیاز است که برنامه نسبت به رویدادهای ماوس و صفحه کلید رفتارهای خاصی داشته باشد. رویدادهای ماوس، به شرح زیر هستند :

- جابه‌جایی ماوس
- کلیک (click)
- دوبارکلیک (double-click)
- کلیک راست (right-click)
- عملیات کشیدن ــ رهاکردن (Drag & Drop)

با استفاده از مشخصهٔ MousePointer می‌توان شکل ظاهری اشاره‌گر ماوس را تغییر داد.

ترتیب رویدادهای ماوس از نظر ایجاد آن‌ها به وسیلهٔ ویندوز چنین است :

۱ــ MouseDown

۲ــ MouseUP

۳ــ Click

۴ــ Dblclick

۵ــ MouseUp

رویدادهای KeyDown ،Keypress و KeyUp مربوط به صفحه کلید هستند.

از مشخصهٔ KeyPreview زمانی استفاده می‌کنیم که بخواهیم فرمی با داشتن یک یا چند کنترل، به رویدادهای صفحه کلید پاسخ دهد. معمولاً از KeyPreview هنگامی استفاده می‌شود که چندین کنترل، نیاز به یک واکنش در برابر رویدادهای کلید داشته باشند.

برای ارسال ضربات کلید به برنامه، از دستور SendKeys استفاده کنید.

خودآزمایی

۱ــ برنامه‌ای بنویسید که بدون استفاده از خاصیت PasswordChar بتواند کلمهٔ رمزی را از ورودی بگیرد و آن را نمایش دهد.

۲ــ برنامه‌ای بنویسید که با هربار زدن یک کلید (حداکثر ۴ بار) یک عکس متفاوت را در کنترل Image نشان دهد و پس از زدن آخرین مرتبه، عکس ثابت بماند.

۳ــ برنامهٔ قبل را به گونه‌ای تغییر دهید که پس از رسیدن به آخرین عکس، دوباره به ابتدا بازگردد.

۴ــ برنامه‌ای بنویسید که یک تصویر را نمایش داده و سپس به وسیلهٔ متد PaintPicture آن را کوچک‌تر کرده و در یک کنترل تصویر دیگر نشان دهد (عمل Zoom Out).

۵ــ برنامه‌ای بنویسید که یک تصویر را به وسیلهٔ کنترل Image نمایش داده و قابلیت بزرگ‌نمایی از ۰/۲۵ تا ۴ برابر را برای آن ایجاد کند.

۶ــ برنامهٔ قبل را فقط با PictureBox ایجاد کنید.

۷ــ برنامه‌ای بنویسید که به وسیلهٔ کلیدهای جهت‌دار (Arrow Key) یک برچسب حاوی نام شما را حرکت دهد. میزان حرکت به وسیلهٔ یک کادر متن تعیین شود.

۸ــ برنامهٔ قبل را به گونه‌ای تغییر دهید که به وسیلهٔ کلیدهای PageUP و PageDowm بتوان میزان حرکت را در هر بار دو برابر یا نصف کرد.

۹ــ یکی از راه‌های ساده برای رمزکردن یک رشته، افزودن یا کاهش کد اسکی یک کاراکتر است. به‌عنوان مثال، اگر به کد اسکی هر کاراکتر ۴ واحد اضافه کنیم، کلمهٔ ALI به EMP عوض می‌شود.

برنامه‌ای بنویسید که به وسیلهٔ یک کادر متن، رشته‌ای را دریافت و در همان لحظهٔ ورود، آن را رمز کند. سپس با زدن دکمهٔ Decode و استفاده از فرمان Sendkeys رشتهٔ کادر متن اول را به کادر متن دیگری بفرستد تا در آن رمزگشایی شود (یعنی به شکل اصلی دیده شود).

توجه : از هیچ تابع رشته‌ای نباید استفاده کنید.

۱۰ــ برنامه‌ای بنویسید که تصویری را نمایش دهد که با کلیک روی تصویر، محل آن به‌طور تصادفی تغییر یابد (از فرم خارج نشود).