فصل دوم

ایجاد منو

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

- **با Application Wizard منوهای ساده‌ای را برای برنامه‌های کاربردی ایجاد کند؛**
- **با Menu Editor منوهای سفارشی و حرفه‌ای را ایجاد کند؛**
- **منوهای بازشو (میانبر) را ایجاد کند که با کلیک راست کاربر در هر جایی از فرم ظاهر می‌شود.**

۱ــ۲ــ منوهای استاندارد ویندوز

هر برنامه‌ای را که بیش از یک عمل انجام می‌دهد می‌توان با افزودن منوهایی کارآمدتر کرد. یکی از هدف‌های طراحی برنامه، ایجاد ویژگی‌هایی است که کاربرد آن را ساده‌تر کند. منویی که به‌طور مؤثر طراحی شده باشد، این هدف را برآورده خواهد کرد.

منو متداول‌ترین ویژگی قابل مشاهده برای برنامهٔ کاربردی است که استفاده از برنامه را ساده‌تر خواهد کرد. با وجود منو، محیط برنامه برای کاربران، طبیعی و آشنا خواهد بود. منویی که به صورت بدون تأثیر (بد) طراحی شده است، سبب سردرگمی کاربران و عدم درک چگونگی کار برنامه می‌شود.

اغلب برنامه‌ها دارای عملیات روی پرونده‌ها هستند که به کاربران امکان می‌دهد پرونده‌ها را ایجاد و ذخیره کنند. برنامه‌هایی که از ویژگی‌های متداول ویندوز مثل بازکردن و ذخیرهٔ پرونده‌ها یا کپی کردن متن استفاده می‌کنند، بهتر است از استانداردهای منوهای ویندوز پیروی کنند.

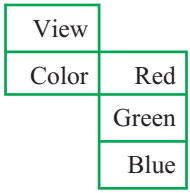
هدف مایکروسافت ایجاد ویژگی‌های استاندارد در کل رابط گرافیکی کاربر است. این بدین معنی است که اغلب برنامه‌های تحت ویندوز ظاهری شبیه هم دارند. به عنوان مثال، روش دسترسی به

گزینهٔ Print در کل برنامه‌ها مشابه هم است. این استانداردها در چندین مورد مثل سیستم‌های راهنما رعایت می‌شوند.

عناصر خاص منوهای ویندوز که از استاندارد ویندوز پیروی می‌کنند، در جدول زیر، فهرست شده‌اند.

جدول ۱ــ۲ــ موارد متداول منوهای استاندارد ویندوز

ویژگی	تعریف
Caption	استفاده از یک یا دو کلمهٔ خاص
Organization	گزینه‌های منو که براساس وظیفه گروه‌بندی خواهند شد تا حداقل تعداد سطوح را برای دسترسی به هر ویژگی ارائه کنند
Access Keys	هر گزینه باید یک کلید دسترسی داشته باشد تا از طریق صفحه کلید نیز بتوان به گزینه‌ها دسترسی داشت در هر قسمت از منو، کلید باید منحصر به فرد باشد و معمولاً حرف اول Caption است
Shortcut Keys	هر گزینه‌ای از منو که نیاز است در بخشی از برنامه قابل دسترس باشد، باید یک کلید میانبر داشته باشد هر کلید میانبر می‌تواند فقط مربوط به یک گزینه باشد
Check Box	گزینه‌هایی که نیاز به فعال و غیرفعال‌شدن دارند، باید یک کادر علامت داشته باشند که نشان‌دهندهٔ حالت آن باشد گزینه‌ای که یک کادر محاوره‌ای را باز می‌کند باید دارای ... در انتهای نام خود باشد



Organization تعیین سر منو و زیر منوها براساس وظیفهٔ آن‌ها می‌باشد. در شکل روبرو view سر منو و دارای سطح صفر Color زیر منوی View و داری سطح یک و گزینه‌های Red، Green و Blue زیر منوهای Color و دارای سطح ۲ هستند.

Accesskey برای دسترسی به گزینه‌ها تعریف می‌شود و سبب می‌شود که به جای استفاده از ماوس برای انتخاب منو از صفحه کلید استفاده نماییم (مانند Alt f برای بازکردن منوی file) ولی Shortcut Key کلید میانبر برای وظیفه‌ای است که در صورت انتخاب منو انجام می‌شود (مانند Ctrl C برای عمل کپی در نرم‌افزار word)

۲ــ۲ــ ایجاد منو

در ویژوال بیسیک برای ایجاد منو دو شیوه وجود دارد که عبارتند از :

- Application wizard
- Menu Editor

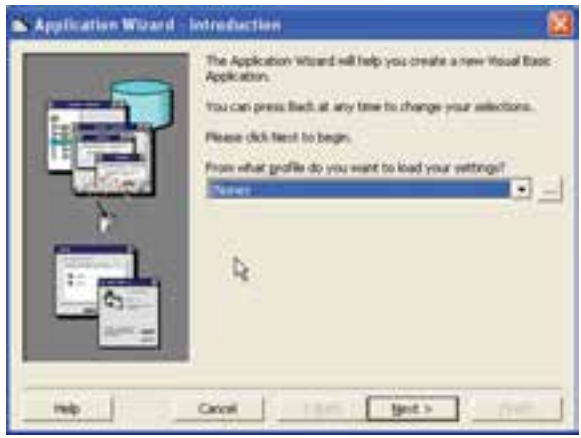
Application Wizard ابزار مفیدی برای ایجاد برنامهٔ کاربردی جدید است. این بدین معنی است که این ابزار برای ایجاد برنامهٔ کاربردی با ویژگی‌های استاندارد مورد استفاده قرار می‌گیرد. مزیت عمدهٔ Application Wizard ایجاد منوهایی است که دارای ویژگی‌های استاندارد ویندوز هستند. به سادگی، می‌توان این ویژگی‌ها را که به صورت الگو ارائه شده‌اند انتخاب کرد. در صورتی که می‌خواهید ویژگی‌های دیگری را به آن‌ها اضافه کنید، باید برنامه‌نویسی کنید یا از Menu Editor استفاده کنید. بعد از کلیک کردن روی Finish در Application Wizard، برنامهٔ اصلی تولید می‌شود که امکان انجام تغییرات با Menu Editor را فراهم می‌کند.

۱ــ۲ــ۲ــ ایجاد یک منوی ساده با Application Wizard

۱ــ از طریق کادر محاوره‌ای پیش‌فرض که هنگام شروع Visual Basic 6.0 باز می‌شود همچنین می‌توانید با انتخاب گزینهٔ New Project از منوی File، برنامهٔ VB Application Wizard را شروع کنید (شکل ۱ــ۲).

شکل ۱ــ۲ــ روی نشانه Application Wizard برای شروع ویزارد دو بار کلیک کنید.

۲ــ کادر محاوره‌ای Introduction امکان استفاده از پروفایل‌های قبلی را که ذخیره کرده‌اید فراهم می‌کند (شکل۲ــ۲). گزینه‌های پیش‌فرض را تغییر ندهید و روی Next کلیک کنید.



شکل ۲ــ۲ــ پروفایل‌ها امکان بارگذاری مجدد گزینه‌ها از پرونده‌های قبلی Application Wizard را فراهم می‌کنند.

۳ــ در کادر محاوره‌ای Interface Type، نوع صفحهٔ آغازین برنامهٔ کاربردی را انتخاب کنید (شکل۳ــ۲). برای این برنامهٔ کاربردی نمونه، Single Document Interface را انتخاب کنید. نام پروژهٔ پیش‌فرض را تغییر ندهید و Next را انتخاب کنید.

شکل ۳ــ۲ــ انتخاب نوع رابط بستگی به چگونگی استفاده از برنامهٔ کاربردی خواهد داشت. برنامه‌های کاربردی چند وظیفه‌ای اغلب از MDI [۱] استفاده می‌کنند.

۱ــ توضیح برنامه‌های MDI در فصل سوم آمده است.

۴ــ یک منوی پیش‌فرض بر اساس استاندارد ویندوز ایجاد می‌شود و شما می‌توانید منو و زیرمنوها را تغییر دهید. بعد از انجام تغییرات، روی Next کلیک کنید (شکل ۴ــ۲).

شکل ۴ــ۲ــ نوع منو و گزینه‌ها را برای برنامهٔ کاربردی انتخاب کنید.

۵ــ Application Wizard امکان سفارشی کردن نوار ابزار، پرونده منبع، مرورگر، اتصال پایگاه داده و سایر الگوها را فراهم می‌کند. برای مثال، از این کادرهای محاوره‌ای صرف‌نظر کرده و پنج بار روی Next کلیک کنید تا به آخرین کادر محاوره‌ای برسید.

۶ــ در آخرین کادر محاوره‌ای، می‌توان پروفایل را ذخیره کرد (شکل ۵ــ۲). نامی را برای پروفایل وارد کنید که مرتبط با برنامهٔ کاربردی باشد. بعد از وارد کردن نام، روی Finish کلیک کنید تا Application Wizard کامل شود. (کلیک کردن روی Finish در کارهای محاوره‌ای قبلی، سبب می‌شود که ویزارد از ذخیرهٔ پروفایل، صرف‌نظر کند.)

شکل ۵ــ۲ــ نام پروفایل را در آخرین صفحه وارد کنید.

نکته

پسوند پروفایل RWP است.

۳ــ۲ــ کاربرد Menu Editor

Menu Editor امکان ایجاد نوار منو یا اصلاح منوهای ایجاد شده را فراهم می‌کند. این منوها به طور معمول در بالای فرم قرار می‌گیرد.

روش‌های باز کردن برنامهٔ Menu Editor عبارت‌اند از :

- فشار دادن کلیدهای Ctrl E
- دکمهٔ Menu Editor در نوار ابزار استاندارد
- انتخاب گزینهٔ Menu Editor از منوی Tools

مثال ۱ــ۲

ایجاد یک منوی ساده

۱ــ پروژهٔ جدیدی را باز کنید.

۲ــ مشخصهٔ Caption فرم را به Simple Menu تغییر دهید.

۳ــ برنامهٔ Menu Editor را باز کنید (توجه کنید که برای باز کردن این برنامه، باید فوکوس روی فرم باشد.)

شکل ۶ــ۲ــ دکمه‌های فلش، امکان تورفتگی و سازماندهی گزینه‌های در لیست منو را فراهم می‌کنند.

۴ــ در کادر محاوره‌ای Menu Editor و در قسمت Caption، عبارت File و در قسمت Name، عبارت mnuFile را وارد کنید.

۵ــ برای ایجاد گزینهٔ این منو روی Next کلیک کنید.

۶ــ در قسمت Caption، عبارت Exit و در قسمت Name، عبارت itmExit را وارد کنید. به دلیل این‌که این منو زیرمنوی (گزینهٔ) منوی File است، روی دکمهٔ فلش راست کلیک کنید (شکل ۶ــ۲).

۷ــ روی Ok کلیک کنید تا منوی ایجاد شده را مشاهده کنید (شکل ۷ــ۲).

۸ــ منوی File را باز کرده و روی Exit کلیک کنید. پنجرهٔ کد با رویداد itmExit_Click() ظاهر می‌شود.

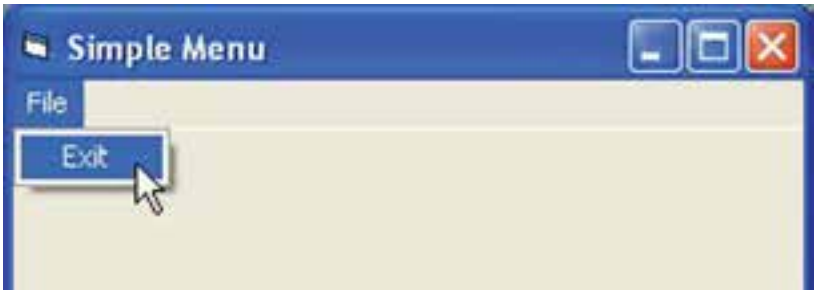
۹ــ دستور Unload Me را به روال رویداد itmExit اضافه کنید (شکل ۸ــ۲).

نکته

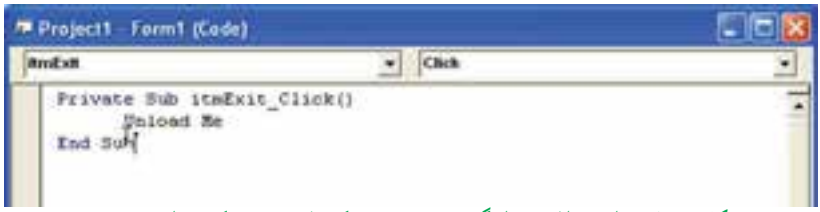
برای بارگذاری فرم در حافظه از دستور loadMe و برای پاک کردن آن از حافظه از دستور Unload Me استفاده کنید.

ـ منظور از Me فرم جاری است.

۱۰ــ کلید F5 را فشار دهید تا کد اجرا شود.



شکل ۷ــ۲ــ یک گزینه در منو شبیه هرکنترل دیگری در ویژوال بیسیک است که دارای مشخصه‌ها و یک رویداد به نام Click است.



شکل ۸ــ۲ــ برای روال رویداد گزینه‌ها مانند سایر کنترل‌های VB، کدی را بنویسید.

۱ــ۳ــ۲ــ تنظیم مشخصه‌های منو : همان‌طور که قبلاً نیز ذکر شد، منو کنترلی با مجموعه‌ای از مشخصه‌ها و یک رویداد ()Click است. جدول ۲ــ۲ متداول‌ترین مشخصه‌های مورد استفاده برای شیء Menu را نشان می‌دهد.

مانند هر کنترلی، می‌توان هنگام ایجاد منو یا گزینهٔ منو، مشخصهٔ Name را مقداردهی کرد. عدم مقداردهی Name سبب بروز خطا می‌شود. عدم مقداردهی مشخصهٔ Caption سبب نمایش یک خط خالی خواهد شد.

۲ــ۳ــ۲ــ اضافه کردن کلیدهای دسترسی به گزینه‌های منو : علاوه بر کلیک کردن روی یک گزینهٔ منو برای اجرای یک وظیفه، می‌توان با استفاده از کلیدهای دسترسی، به گزینه‌ها دسترسی پیدا کرد. کلیدهای دسترسی به کاربران امکان می‌دهند گزینه‌ها را با فشار دادن Alt و سپس حرف تعیین شده انتخاب کنند. بعد از این‌که منو باز شد، کاربران می‌توانند با فشار دادن کلید دسترسی، گزینه را انتخاب کنند.

جدول ۲ــ۲ــ مشخصه‌های متداول شیء Menu

مشخصه	مقدار / نوع	شرح
Caption	String	متنی که روی نوار منو ظاهر می‌شود
Checked	Boolean	یک علامت ✓ قبل از رشتهٔ Caption گزینه قرار می‌دهد
Enabled	Boolean	درصورتی که True باشد، رشتهٔ Caption به صورت خاکستری و غیرفعال نخواهد بود
Name	String	نام شیء فقط در زمان طراحی قابل دسترس است
Shortcut	N/A	یک ترکیب کلیدی می‌باشد که به‌عنوان کلید میانبر برای وظیفه‌ای که این گزینه انجام می‌دهد، به‌کار می‌رود این مقدار را فقط در زمان طراحی و از طریق فرمتی که در کادر لیست Shortcut در Menu Editor ظاهر می‌شود، می‌توانید انتخاب کنید
WindowList	Boolean	یک منوی سطح بالا در فرم MDI ایجاد می‌کند تا لیستی از پنجره‌های باز را نمایش دهد [۱]

۱ــ فرم MDI می‌تواند همزمان چند پنجرهٔ باز داشته باشد (مانند نرم‌افزار word) لیست پنجره‌های باز در منویی که خاصیت w ndow L st آن true است دیده می‌شود.

برای تعریف یک کلید دسترسی، در کادر متن Caption، قبل از نویسهٔ موردنظر، از نویسهٔ & استفاده کنید (شکل ۹ــ۲).

شکل ۹ــ۲ــ کلیدهای میانبر و دسترسی را قبل از کامپایل کردن برنامه تعیین کنید.

نکته

از دو کلید دسترسی مشابه در یک بخش منو استفاده نکنید.

کلیدهای دسترسی مطابق با منوها گروه‌بندی می‌شوند و در صورتی می‌توانید از حرف یکسان برای گزینه‌ها استفاده کنید که در منوهای مختلف ظاهر شوند. اگر دو کلید دسترسی مشابه در یک بخش منو قرار دهید، اولین کلید در سلسله مراتب ابتدا اجرا خواهد شد. هنگامی که کلید برای دومین بار فشار داده شود، گزینهٔ بعدی با همان کلید، انتخاب می‌شود.

۳ــ۳ــ۲ــ اضافه کردن کلیدهای میانبر به گزینه‌ها: از طریق Application Wizard نمی‌توان برای گزینه‌ها کلیدهای میانبر اضافه کرد. ولی انجام این کار در Menu Editor ممکن است. استفاده از کلیدهای میانبر، روش دیگری برای اجرای وظایف گزینه‌ها از طریق صفحه کلید است. کلید میانبر مربوط به هر منو را می‌توان در کادر محاوره‌ای Menu Editor از لیست Shortcut انتخاب کرد (شکل ۱۰ــ۲).

شکل ۱۰ـ۲

نکته

هر برنامهٔ کاربردی فقط می‌تواند یک نمونه از کلید میانبر را داشته باشد. به‌عنوان مثال، اگر از Ctrl+N برای New استفاده کرده باشید، Ctrl+N را برای گزینهٔ Open به‌کار نبرید. هنگام تعیین کلید میانبر تکراری، VB خطایی را نشان می‌دهد (شکل۱۱ـ۲).

شکل ۱۱ـ۲

نکته

برای گروه‌بندی گزینه‌های یک منو نیاز به خط جداکننده داریم. برای قراردادن این خطاها در بین گزینه‌های منو، در قسمت Caption از کادر محاوره‌ای Menu Editor، خط تیره قرار دهید و نام گزینه را با پیشوندی مثل Sep شروع کنید. توجه داشته باشید که Name خط جداکننده را خالی رها نکنید.

۴ــ۳ــ۲ــ ایجاد منوهای بازشو : دو نوع منو وجود دارد : نوار منویی و منوی بازشو.

نوار منویی به طور ثابت در بالای فرم قرار می‌گیرد اما **منوی بازشو** منویی است که در هر جایی از فرم ظاهر می‌شود.

منوی بازشو در موارد مورد نیاز مانند کلیک راست در محل اشاره‌گر ماوس ظاهر می‌شود. برای ایجاد چنین منوهایی نیز از Menu Editor استفاده می‌شود. در حقیقت، در مواقع مورد نیاز می‌توانیم قسمتی از منوی تعریف شده به‌وسیلهٔ Menu Editor را در محل اشاره‌گر یا در محل مورد نظر فعال کنیم. در VB با استفاده از دستور PopupMenu می‌توان قسمتی از منوها را به همراه زیر منوهای بعد از آن در ناحیه‌ای از فرم ظاهر کرده و از آن استفاده نمود.

شکل کلی این دستور به صورت زیر است :

```
PopupMenu MenuName [Xpos],[Ypos]
```

در این ساختار MenuName نام منویی است که می‌خواهیم در فرم نمایش داده شود. هم‌چنین دو پارامتر اختیاری Ypos و Xpos، مختصات محل ظاهر شدن منو را تعیین می‌کنند. اگر از این دو مقدار استفاده نشود VB به‌طور خودکار از مختصات محل فعلی ماوس به جای این دو متغیر استفاده می‌کند.

به عنوان مثال، دستور زیر سبب می‌شود تا منوی Edit، در محل فعلی ماوس نمایش داده شود :

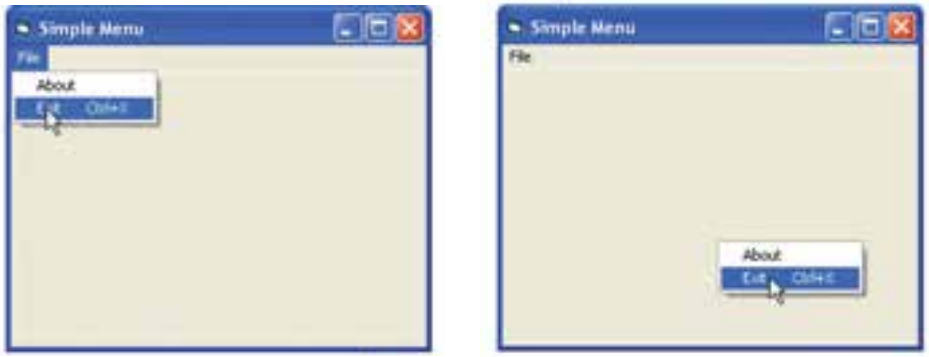
```
PopupMenu   MenuEdit
```

می‌توانید نمونهٔ این نوع منوها را تقریباً در همه جای ویندوز مشاهده کنید. در هر قسمت از ویندوز اگر کلیک راست کنید معمولاً منویی ظاهر شده و گزینه‌هایی را ارائه می‌کند. طریقهٔ ایجاد چنین منوهایی را در مثال فوق آموختید، ولی ممکن است سؤال کنید که : وقتی روی یک فرم کلیک می‌شود، برنامه چگونه می‌تواند تشخیص دهد که کدام یک از کلیدهای ماوس فشار داده شده است؟ نوع کلید ماوس را با بررسی آرگومان ورودی Button در رویدادهای MouseDown و MouseUp می‌توان تعیین کرد.

در صورتی که بخواهید می‌توانید یک فرمان را هم در نوار منویی و هم منوی بازشو قرار دهید (شکل ۱۲ــ۲). کد زیر چگونگی انجام این کار را شرح می‌دهد.

```
Private Sub Form  MouseDown (Button As Integer, shift As Integer, X
As  Single,Y As Single)
    If Button   2 Then
        PopupMenu mnuFile
    End If
End Sub
```

برای این‌که این کد کار کند، نیاز به ایجاد فرمی دارید که شامل یک کنترل Menu به نام mnuFile باشد که حداقل دارای یک گزینه است.

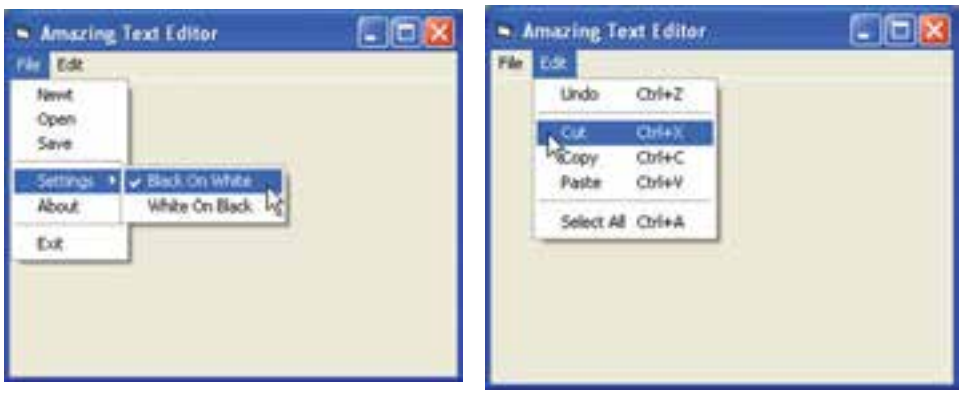


شکل ۱۲ــ۲ــ هنگامی که از متد PopupMenu استفاده می‌کنید، فقط گزینه‌های منو ظاهر خواهند شد.

پژوهش: اگر سر منویی غیرفعال باشد، زیر منوهای آن به چه صورتی مشاهده می‌شوند؟

مثال ۲ــ۲

ایجاد منوهای پیچیده : منویی ایجاد کنید که گزینه‌های زیر در آن وجود داشته باشد :



شکل ۱۳ــ۲

قبل از شروع طراحی، باید ویژگی‌های برنامه را مرور کنید تا سیستم منو را به طور مناسب طراحی و گروه‌بندی کنید. اغلب، نوارهای منو با منوی File و Edit شروع می‌شوند.

File Menu	Edit Menu
New	Undo Ctrl + Z
Open	Cut Ctrl +X
Save	Copy Ctrl +C
Settings	Paste Ctrl +V
About	Select All Ctrl +A
Exit Ctrl + E	

به گروه‌بندی سیستم منو توجه کنید. می‌توان این منو را در Menu Editor پیاده سازی کرد.

جدول ۳ــ۲ سلسله مراتب منو و مشخصه‌های Name ، Caption و کلیدهای میانبر برای هر شیء منو را نشان می‌دهد.

جدول ۳ــ۲ــ شیء‌های منو برای مثال ۲ــ۲

Name	Caption	Level	Shortcut
mnuFile	& File	0	None
itmNew	& New	1	None
itmOpen	& Open	1	None
itmSave	& Save	1	None
sepOne	–	1	None
itmSettings	Se&ttings	1	None
itmBlackOn White	Black On White	2	None
itmWhiteOnBlack	White On Black	2	None
itmAbout	&About	1	None
sepTwo	–	1	None
itmExit	E&xit	1	Ctrl + x
mnuEdit	&Edit	0	None
itmUndo	&Undo	1	Ctrl + z
sep Three	–	1	None
itmCut	Cu&t	1	Ctrl + x
itmCopy	&Copy	1	Ctrl + C
itmPaste	&Paste	1	Ctrl + V
sepFour	–	1	None
itmSelectAll	Select&All	1	Ctrl+A

۴ــ۲ــ شیء Clipboard

یکی از ویژگی‌های مهم سیستم عامل ویندوز، قابلیت انتقال داده‌ها از یک برنامهٔ کاربردی به برنامهٔ کاربردی دیگر از طریق حافظهٔ Clipboard است. تمام برنامه‌های کاربردی امکان دسترسی به این حافظه را دارند. می‌توان هر شیء موجود در ویندوز (از متن ساده تا یک تصویر) را در این حافظه ذخیره کرد.

در VB می‌توان از طریق شیء Clipboard به این حافظه دسترسی داشت. این شیء به‌طور پیش‌فرض در پروژهٔ Standard EXE قرار دارد و می‌توان از آن استفاده کرد. شیء Clipboard هیچ مشخصه‌ای ندارد ولی دارای چندین متد است. برخی از متدهای Clipboard در جدول ۴ــ۲ ذکر شده است.

جدول ۴ــ۲ــ متدهای شیء Clipboard

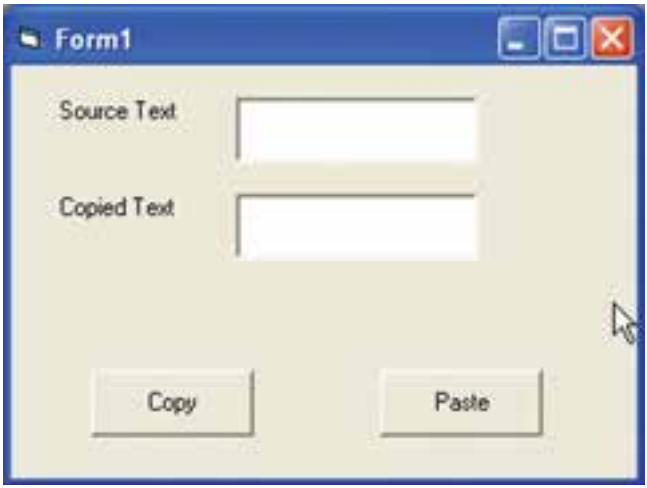
متد	شرح
Clear	تمام داده‌ها را از حافظهٔ Clipboard پاک می‌کند
GetText	متن ASCII را از حافظهٔ Clipboard بازیابی می‌کند
SetText	متن ASCII را به حافظهٔ Clipboard ارسال می‌کند

مثال ۳ــ۲

فرمی ایجاد کنید که شامل دو کادر متن به نام‌های txtMain و txtClip و دو دکمهٔ فرمان به نام‌های cmdCopy و cmdPaste باشد. می‌خواهیم هر چیزی که در txtMain نوشته می‌شود، با انتخاب دکمهٔ Copy به حافظهٔ Clipboard کپی شده و با انتخاب Paste از حافظهٔ Clipboard به کادر متن txtClip کپی شود (شکل ۱۴ــ۲).

کد دکمه‌های فرمان این مثال به صورت زیر خواهد بود :

```
Private Sub cmdCopy   Click()
    Clipboard.SetText(txtMain.Text)
End Sub
Private Sub cmdPast  Click()
    txtCLip.Text   Clipboard.GetText
End Sub
```



شکل ۱۴ــ۲

در این مثال، کل متن موجود در txtMain با انتخاب دکمهٔ Copy به حافظهٔ Clipboard وارد می‌شود. در صورتی که می‌خواهید فقط قسمتی از متن که انتخاب کرده‌اید به حافظهٔ Clipboard وارد شود، در روال مربوط به دکمهٔ Copy به جای Text از مشخصهٔ SelText استفاده کنید :

```
Clipboard.SetText(txtMain.SelText)
```

نکته

برای انتخاب قسمتی از متن در کادر متن در زمان اجرا، روی نویسهٔ اول کلیک کنید و تا نویسهٔ مورد نظر درگ کنید. انتخاب متن به‌صورت کدنویسی نیز ممکن است. برای انجام این کار، از دو مشخصهٔ SelStart و SelLenght استفاده کنید. SelStart محل شروع انتخاب و SelLenght طول نویسه‌های انتخاب شده را مشخص می‌کنند. به‌عنوان مثال، می‌خواهیم کلمه‌ای را که در کادر متن قرار دارد از نویسهٔ دوم تا پنجم انتخاب کرده و به حافظهٔ Clipboard منتقل کنیم.

```
Text1. Selstart = 1
Text1. SelLenght = 4
Clipboard. SetText (Text 1. SelText)
```

خودآزمایی

۱ــ در چه مواقعی از منوها استفاده می‌شود؟

۲ــ در کادر Menu Editor، گزینهٔ Name به چه منظوری مورد استفاده قرار می‌گیرد؟

۳ــ انواع منوها را نام ببرید.

۴ــ وقتی کاربر از کلید میانبر استفاده می‌کند، کدام رویداد تحریک خواهد شد؟

۵ــ فرمی ایجاد کنید که اگر روی آن کلیک راست کنید، منویی باز شود و امکان تغییر رنگ فرم به رنگ‌های قرمز، سبز و آبی را فراهم کند.

۶ــ در تمرین ۵، TextBox به فرم اضافه کرده و منویی برای تغییر رنگ متن آن به رنگ‌های زرد و آبی فراهم کنید.

۷ــ به مثال ۳ــ۲ دکمهٔ cut را اضافه کنید و به جای انتقال محتوای txtMain تنها قسمت انتخاب‌شده را cut و past کنید.

فصل سوم

خطایابی و اشکال‌زدایی برنامه

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

- **اصول کار با دستور On Error Goto را بیان کرده و در برنامه‌ها آن را به‌کار ببرد؛**
- **نحوهٔ کار باشیء ERR را شرح داده و در برنامه‌های خود از آن استفاده کند؛**
- **با دستور Resume کار کند و اجرای برنامه‌های قطع شده را از سر بگیرد؛**
- **برنامه‌های خود را اشکال‌زدایی کند.**

۱ــ۳ــ انواع خطاها در برنامه‌نویسی

در برنامه‌نویسی سه نوع خطا وجود دارد :

۱ــ خطای نحوی (syntax error) : شامل خطا در املاء یا محل قرارگیری کلیدواژه‌ها می‌باشد مانند استفاده از دستورIf بدون کلیدواژهٔ then در زبان بیسیک

```
If  a>b
Print a
```

دستور بالا دارای خطای نحوی "Expected: then or Go To" (فاقد then یا GoTo) است.

این خطاها در زمان کامپایل برنامه مشخص می‌شود.

۲ــ خطای زمان اجرا (Runtime error) : خطاهایی که هنگام اجرای دستور رخ می‌دهد و تا قبل از اجرای آن مشخص نمی‌شود.

مثال ۱ــ۳

خطای تقسیم بر صفر

```
i  0  : a  5
b  a/i
```

این نوع خطاها در زمان کامپایل قابل تشخیص نیست.

۳ــ خطای منطقی (logical error) : خطاهای منطقی در زمان کامپایل و اجرا مشکلی ایجاد نمی‌کند بلکه از نظر منطقی خروجی برنامه، خروجی مورد نظر نیست و برنامه‌نویس در پیاده‌سازی منطق برنامه و حل مسأله اشتباه کرده است. بنابراین نتیجه مطلوب حاصل نمی‌شود.

```
for   i  2      to 10
        print  i ;
Next
```

هدف تکه کد بالا تولید اعداد زوج کوچکتر از ۱۲ بوده است ولی خروجی آن عبارتست از

2 3 4 5 6 7 8 9 10

زیرا برنامه‌نویس step 2 را فراموش کرده است و باید کد به‌صورت زیر اصلاح شود.

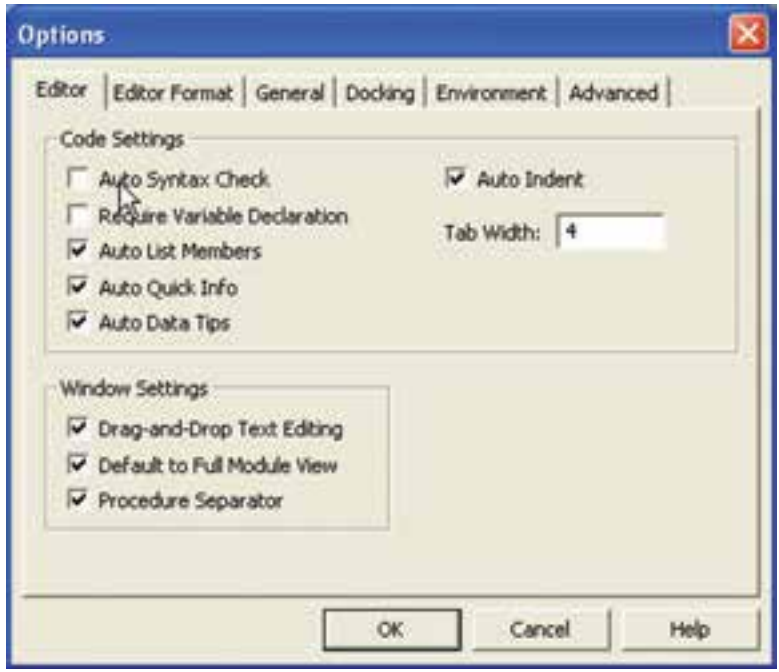
```
for    i  2    to  10   step 2
        print  i ;
Next
```

۲ــ۳ــ رفع اشکال متغیرهای اعلان نشده با Option Explicit

همزمان با کدنویسی، IDE ویژوال بیسیک خطاهای نحوی را که به‌وجود می‌آیند اعلام می‌کند (شکل ۱ــ۳). خطاهای نحوی شامل املاء یا محل قرارگیری کلیدواژه‌ها هستند. به این نوع خطاها می‌توان به سادگی پی برد و آن‌ها را رفع کرد.

شکل ۱ــ۳ــ IDE ویژوال بیسیک دستورات IF بدون کلیدواژهٔ Then را هنگام تایپ شناسایی می‌کند. ولی اگر فقط کلیدواژه‌های End If اشتباه باشند، هنگام کامپایل کد، اعلام خواهد شد.

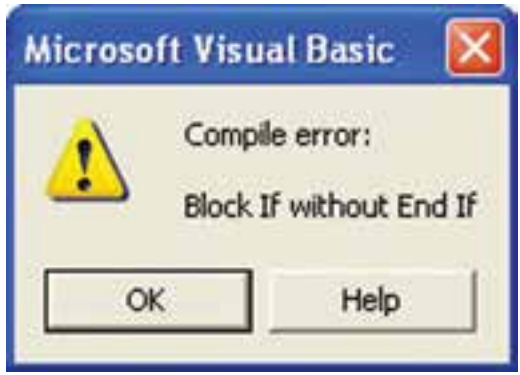
همزمان با تایپ، VB خطاهای نحوی را اعلام می‌کند. در صورتی که می‌خواهید VB کار را قطع نکند و برای هر خطایی، پیغامی نمایش ندهد، از منوی Tools گزینهٔ Options را انتخاب کرده و کادر علامت Auto Syntax Check را پاک کنید (شکل ۲ــ۳).



شکل ۲ــ۳ــ غیرفعال کردن Auto Syntax Check سبب می‌شود که پیغام‌های خطای نحوی را هنگام تایپ مشاهده نکنید.

پاک کردن این کادر علامت سبب می‌شود که کامپایلر از پیدا کردن خطاها در هنگام تایپ، صرف‌نظر کند.

هنگامی که کد را درون IDE اجرا می‌کنید، ویژوال بیسیک خطاهایی مثل نوع اشتباه و بلاک‌های کد کامل نشده را گزارش می‌کند (شکل ۳ــ۳). فقط در صورتی که Option Explicit تنظیم شده باشد، ویژوال بیسیک کد متغیرهای اعلان نشده را اجرا می‌کند. هنگامی که کلید واژهٔ Option Explicit را در بخش General یک فرم یا مدول وارد می‌کنید، همهٔ متغیرهای کد باید با استفاده از کلیدواژه‌های Public، Private، Dim یا Static به‌طور صریح اعلان شوند.



شکل ۳ــ۳ــ IDE هنگام کامپایل کد، یک بلاک حلقهٔ کامل نشده را گزارش می‌کند.

اشتباهات تایپی ساده منجر به بروز خطاهای بزرگی در کد می‌شوند. به‌عنوان مثال، در کد زیر متغیری با نام intMyNum اعلان شده است ولی در خط ۴ از intMyNim استفاده شده است. به دلیل این که Option Explicit استفاده نشده است، VB به‌طور خودکار متغیر جدیدی به نام intMyNim ایجاد کرده و آن را با صفر مقداردهی می‌کند. (خطای منطقی) شکل ۴ــ۳ نتیجه را نشان می‌دهد.

```
01 Private Sub cmdUnWit  Click()
02      Dim intMyNum As Integer
03       intMyNum    2  2
04      MsgBox CStr(intMyNim)
05 End Sub
```

شکل ۴ــ۳ــ اگر از Option Explicit استفاده کرده باشید، پیام خطایی را دریافت خواهید کرد که مشخص می‌کند متغیر intMyNim تعریف نشده است.

۳ــ۳ــ بررسی قطعات کد با BreakPoint

می‌توان کد ویژوال بیسیک را در هر نقطه‌ای از اجرا متوقف کرده و آن را با نقاط قطع (breakpoints) بررسی کرد. یک نقطهٔ قطع، محلی در کد است که می‌توان کد را در طول اجرا متوقف کرد. نقطهٔ قطع را می‌توان به چهار روش تعیین کرد :

- کلیک روی خط کد موردنظر و فشار دادن کلید F9
- کلیک روی آیکن Toggle Breakpoint در نوار ابزار Debug
- انتخاب Toggle Break Point از منوی Debug
- کلیک در حاشیهٔ پنجرهٔ Code

نکته

برای پاک کردن تمام نقاط قطع در کد، از منوی Debug گزینهٔ Clear All Breakpoints را انتخاب کنید. همچنین می‌توان تمام نقاط قطع را با فشار دادن کلیدهای Ctrl+Shift+F9 پاک کرد.

هنگامی که یک نقطهٔ قطع را تعیین می‌کنید، توجه کنید که به رنگ قرمز تبدیل می‌شود. هنگامی که کد قطع می‌شود، خط موردنظر کد به رنگ زرد تبدیل می‌شود. همچنین یک فلش در حاشیهٔ چپ پنجرهٔ Code به خط موردنظر اشاره می‌کند (شکل ۵ــ۳).

شکل ۵ ــ۳ــ برای رجوع به نقطهٔ قطع بعدی، کلید F5 را فشار دهید.

همهٔ اشکالات براثر نحو کد به وجود نمی‌آیند و اغلب اشکالات که سبب بروز خطا می‌شوند به دلیل منطق کد یا اشکال طراحی است. پیدا کردن این نوع اشکالات، مشکل است. از نقاط قطع می‌توانید برای محدود کردن کدی که سبب بروز اشکال شده است، استفاده کنید. بعد از تعیین خطی از کد که می‌دانید اشکال رخ داده است، نقطهٔ قطع را تعیین کرده و ناحیهٔ بروز خطا را با استفاده از watches جست‌وجو کنید.

۴ــ۳ــ نمایش مقادیر متغیرها با Watches

با نگاه دوباره به کد و شکل ۳ــ۳ مشاهده می‌کنیم که دارای اشکال است و دلیل آن نیز روشن است. فرض کنید که دلیل بروز این خطا را نمی‌دانید لذا فرصت مناسبی است که از نقاط قطع و watches استفاده کنید.

۱ــ نقطهٔ قطع را برای خطی از کد که کادر پیام را نشان می‌دهد تعیین کنید.

۲ــ برنامه را اجرا کنید.

۳ــ اشاره‌گر ماوس را روی متغیری که می‌خواهید مقدار آن را مشاهده کنید، درگ نمایید و چند لحظه نگه دارید. پنجرهٔ کوچکی با مقدار متغیر ظاهر می‌شود.

در پنجرهٔ watches از منوی view می‌توان تغییر مقادیر یک متغیر را مشاهده کرد. برای اضافه کردن متغیرهای دیگری به پنجرهٔ watch مراحل زیر را انجام دهید :

۱ــ یک یا دو نقطهٔ قطع برای متغیرهای مورد نظر تعیین کرده و کلید F5 را فشار دهید.

۲ــ در هر نقطهٔ قطع، متغیر مورد نظر را های‌لایت (مشخص) کنید.

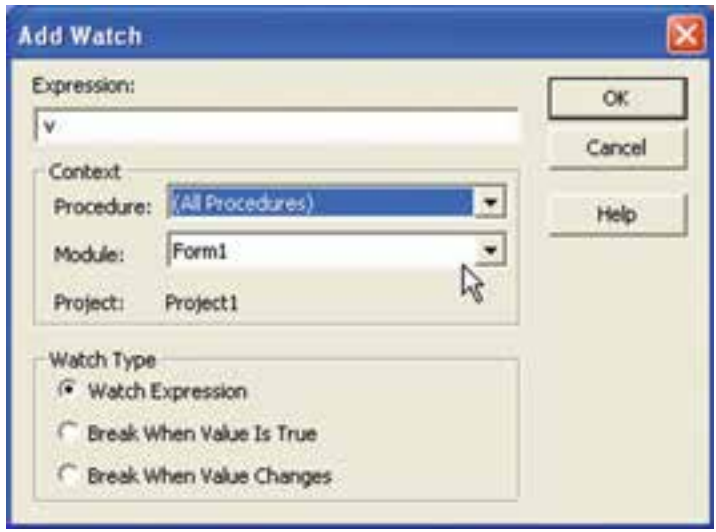
۳ــ کلیک راست کرده و Add watch را از منوی میانبر انتخاب کنید.

۴ــ تنظیمات مناسب را در کادر محاوره‌ای Add watch انجام دهید (شکل۶ــ۳) و سپس روی Ok کلیک کنید.

در پنجرهٔ Watches، قسمت Expression نام متغیر یا مشخصه‌ای نوشته می‌شود که می‌خواهیم مقدار آن را بررسی کنیم. در قسمت Context نام رویداد یا مدولی را که می‌خواهیم مقدار عبارت را در آن بررسی کنیم، می‌نویسیم. در سومین بخش یعنی Watch Type می‌توانید یکی از سه گزینه را انتخاب کنید. با انتخاب اولین گزینه، هنگام رسیدن به نقطهٔ توقف، مقدار Expression را می‌توان در پنجرهٔ Watch مشاهده کرد. انتخاب گزینهٔ دوم سبب می‌شود که در صورت True یا غیرصفر بودن مقدار متغیر کادر Expression، یک نقطهٔ توقف در دستور بعد از آن ایجاد شود. چنانچه گزینهٔ سوم

را انتخاب کنید، اگر مقدار عبارت کادر Expression تغییر کند، یک نقطهٔ توقف در دستور بعدی ایجاد می‌شود.

۵ ــ برای نمایش پنجرهٔ watches (شکل ۷ــ۳)، ازمنوی View گزینهٔ watch window را انتخاب کنید. هنگام اضافه کردن یک watches نیز این پنجره ظاهر می‌شود.



شکل ۶ــ۳ــ کادر محاوره‌ای Add Watch یک ابزار قوی و دارای قابلیت انعطاف است.

شکل ۷ــ۳ــ در حالت Break، مطمئن باشید در محلی از کد که در میدان دید متغیرهای پنجرهٔ watches است، قرار دارید.

پی‌گیری چندین متغیر مستقل که به‌طور پیوسته تغییر می‌یابند می‌تواند خیلی مشکل باشد. پنجرهٔ watches ابزار مؤثری برای پی‌گیری مقادیر در عملیات پویا مثل حلقه‌ها یا آرایه‌ها است.

۱ــ۴ــ۳ــ بررسی خط به خط کد با Step Into و Step Over : می‌توان کد یک برنامه را به صورت خط به خط اجرا و بررسی نمود و اگر خطایی در هر خط وجود دارد آن را رفع کرد. اجرای گام به گام به دو روش انجام می‌شود : Step Into و Step Over.

Step Into را می‌توان با روش‌های زیر اجرا کرد :

- انتخاب گزینهٔ Step Into از منوی Debug
- فشاردادن کلید F8
- کلیک روی نشانهٔ Step Into از نوار ابزار Debug

هنگام استفاده از روش Step Into، اگر در خطی از کد یک روال دیگری فراخوانی شود، اجرای گام به گام وارد روال جدید خواهد شد.

اگر کد را به صورت اجرای گام به گام شروع کنید، ممکن است بعضی مواقع از این‌که هیچ چیزی رخ نداده است متعجب شوید. در صورتی که هیچ کدی در روال رویداد ()Form_Load نداشته باشید، هیچ رویدادی اجرا نمی‌شود. به خاطر داشته باشید که ویندوز یک سیستم عامل رویدادگرا است و رویدادی باید برای کد رخ دهد تا اجرا شود. تنها کدی که به صورت پیش فرض شروع به کار می‌کند، کد مربوط به روال‌های ()Form_Initialize، ()Form_Load یا ()Sub Main است. اگر هیچ کاری داخل این روال‌ها نباشد، برنامه تا زمانی که رویدادی رخ ندهد، هیچ کاری انجام نمی‌دهد.

برای اجرای گام به گام کد، بهتر است یک نقطهٔ قطع در خط موردنظر کد قرار داده و سپس کد را به صورت عادی اجرا کنید.

هنگامی که اجرا به نقطهٔ قطع برسد، اجرای گام به گام شروع خواهد شد. برای خروج سریع از روالی که به صورت گام به گام اجرا می‌شود، باید از Step Out استفاده کرد.

Step Out را می‌توان با روش‌های زیر اجرا کرد :

- انتخاب گزینهٔ Step out از منوی Debug
- فشاردادن کلید Ctrl Shift F8
- کلیک روی نشانه Step Out از نوار ابزار Debug

هنگامی که از روش Step Over برای اجرای گام به گام کد استفاده می‌کنید، در صورت برخورد با خطی که روال دیگری را فراخوانی می‌کند، وارد روال فراخوانی شده نخواهید شد و فقط این خط

کد به صورت یک خط منفرد اجرا خواهد شد.

Step Over را می‌توان با روش‌های زیر اجرا کرد :

- انتخاب گزینهٔ Step Over از منوی Debug
- فشاردادن کلید F8 Shift
- کلیک روی آیکن Step Over از نوار ابزار Debug

۲ــ۴ــ۳ــ متوقف‌کردن خطوط انتخاب شده : هر بار که یک نقطهٔ قطع را تعیین کنید، تا زمانی که به کد برنگشته و آن نقطهٔ قطع یا همهٔ نقاط قطع را پاک نکرده باشید، در کد باقی می‌ماند. تعداد زیاد نقاط قطع، سبب کند شدن سرعت اشکال‌زدایی می‌شود. این کار را می‌توان با استفاده از Run to Cursor ساده‌تر انجام داد تا کد را در نقاط تعیین شده قطع کند.

Run to Cursor را می‌توان با روش‌های زیر اجرا کرد :

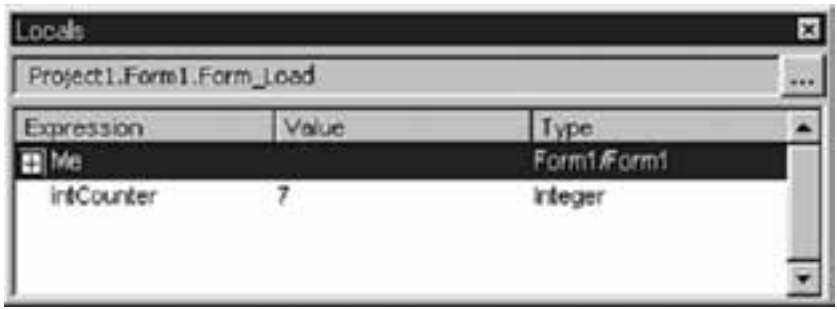
- انتخاب گزینهٔ Run to Cursor از منوی Debug
- فشاردادن کلید F8 Ctrl

روی خطی از کد که می‌خواهید متوقف شود، کلیک کرده و سپس با یکی از روش‌های فوق، Run to Cursor را اجرا کنید. کلید F5 را فشار دهید تا کد اجرا شود. IDE ویژوال بیسیک، اجرای کد را در خطوطی که کلیک کرده‌اید، قطع می‌کند.

۵ــ۳ــ استفاده از ابزارهای اشکال‌زدایی پیشرفته

علاوه بر مشاهدهٔ مقادیر متغیرها و تکنیک‌های اجرای گام به گام، می‌توان از ابزارهای نشان داده شده در شکل‌های ۸ــ۳ تا ۱۱ــ۳ استفاده کرد. ویژوال بیسیک این ابزارها را برای اشکال‌زدایی پیشرفته ارایه می‌کند.

پنجرهٔ Locals (شکل ۸ــ۳) روش ساده‌ای برای مشاهدهٔ تمام متغیرهای موجود در میدان دید جاری است. متغیرها به همراه مقادیرشان در این پنجره فهرست می‌شوند. شیءها دارای یک علامت جمع هستند که می‌توانید روی آن کلیک کنید تا مشخصه‌های شیء را مشاهده کنید. اگر مشخصه، خود شیء دیگری باشد، علامت جمع دیگری را مشاهده خواهید کرد. در این پنجره می‌توان تمام متغیرها را به طور هم زمان و بدون کلیک‌کردن روی هر کدام، مشاهده کرد. برای دسترسی به این پنجره، از منوی View گزینهٔ Locals Window را انتخاب کنید.



شکل ۸ــ۳ــ پنجرهٔ Locals

از پنجرهٔ Immediate : (شکل ۹ــ۳) می‌توان برای آزمایش خطوطی از کد بدون اجرای برنامه استفاده کرد. برای آزمایش این پنجره، کد زیر را در آن وارد کنید :

```
Print 2*3
```

دستور Print مقدار ۶ را ارایه می‌دهد.

شکل ۹ــ۳ــ پنجرهٔ Immediate امکان تایپ کد و اجرای آن را با فشاردادن کلید Enter فراهم می‌کند.

پنجرهٔ Immediate برای دستورهای یک خطی، مناسب است. در این پنجره نمی‌توان متغیر جدیدی را اعلان کرد، ولی می‌توان از هر متغیری که در میدان دید است، استفاده کرد. به عنوان مثال، اگر برنامه در یک روالی قطع شده است که متغیر intCounter در آن تعریف شده است، می‌توان خط

زیر را در پنجرهٔ Immediate تایپ و مقدار آن را مشاهده کرد :

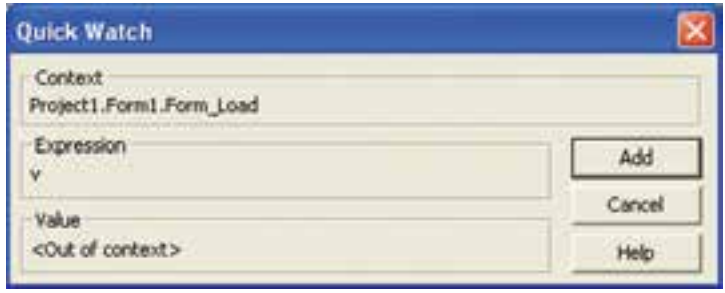
```
Print intCounter
```

همچنین می‌توان مقادیر را در پنجرهٔ Immediate تغییر داده و متدهایی را روی شیءها اجرا کرد. هر چیزی که نیاز به یک خط کد دارد را در این پنجره می‌توان اجرا کرد.

کادر محاوره‌ای Call Stack در صورتی که از چندین روال و رویداد استفاده می‌کنید، مفید است. این کادر محاوره‌ای (شکل ۱۰ــ۳)، تمام روال‌ها و توابع فعال را نشان می‌دهد. عنصر لیست شده در بالای کادر محاوره‌ای، روال جاری است و خط زیر آن خط فراخوانی آن است و الی آخر. این کادر محاوره‌ای با کلیک کردن روی دکمهٔ  از نوار ابزار Debug یا فشاردادن کلیدهای Ctrl L باز می‌شود. لازم به ذکر است که این عمل فقط در حالت قطع ممکن است.

شکل ۱۰ــ۳ــ کادر محاوره‌ای Call Stack

برای نشان‌دادن **کـادر محاوره‌ای Quick Watch** (شکل ۱۱ــ۳) یک قطع را در کـد قرار دهید، روی متغیری کلیک کنید یا عبارتی را مشخص (هایلایت) کرده و از منوی Debug گزینهٔ Quick Watch را انتخاب کنید یا از نوار ابزار Debug روی آیکن  کلیک کنید یا کلیدهای Shift F9 را فشار دهید.



شکل ۱۱ــ۳ــ کادر محاوره‌ای Quick Watch

۶ــ۳ــ کاربرد Find and Replace

در کتاب بسته‌های نرم افزاری ۱ و در بخش آموزش Word با ابزاری به نام Find and Replace آشنا شدید که رشته‌ای را در متن سند جست‌وجو کرده و رشتهٔ دیگری را جایگزین آن می‌کند. در پنجرهٔ کد ویژوال بیسیک نیز می‌توان از این ویژگی استفاده کرد که چگونگی انجام این کار را قبلاً آموخته‌اید.

۷ــ۳ــ ایجاد یک مدیر خطا

اگر نرم‌افزاری داشته باشید که در طی اجرا خراب می‌شود، احتمالاً برنامه نویس آن بخشی از برنامه را فراموش کرده است : مدیریت خطا. هر برنامه‌ای نیاز دارد که به خطاهایی که در درون خودش رخ می‌دهند، پاسخ دهد. در این قسمت، مشاهده می‌کنید که چگونه از قابلیت‌های مدیریت خطا در ویژوال بیسیک استفاده کنید.

برای درک موضوع مثال زیر را انجام می‌دهیم :

پروژهٔ جدیدی ایجاد نموده و فرمی به صورت زیر طراحی کنید :

```
Private sub Start  Click()
        Print 200/i
endsub
```

حال این پروژه را اجرا کنید. به دلیل عدم تعریف متغیر i، این متغیر از نوع variant و مقدار آن صفر خواهد بود. بنابراین خطای تقسیم بر صفر رخ خواهد داد (شکل ۱۲ــ۳).

شکل ۱۲ــ۳

با انتخاب end در کادر محاوره‌ای شکل فوق، اجرای برنامه قطع می‌شود تا خطا رفع شود و با انتخاب Debug اجرای برنامه موقتاً قطع می‌شود و به خط شامل خطا رفته و آن را با رنگ زرد مشخص می‌کند تا خطا را رفع کنید. برنامهٔ فوق را می‌توان به صورت زیر تغییر داد :

```
Dim h As Integer
Private Sub Start  Click()
        On Error Goto TestError
        Print 200/i
        Print "No Problem"
        Exit Sub
        Test Error:
            h    MsgBox ("Error" & Str(Err. Number) &  "  " &
            Err.Description, VbAbortRetryIgnore)
            If h    VbIgnore Then Resume Next
            If h    VbAbort Then End
            If h    VbRetry Then
                i    10
                Resume
            End If
End Sub
```

به دلیل اینکه متغیر i مقداردهی نشده، لذا دارای مقدار صفر است و با اجرای خط ۴ این برنامه خطای تقسیم بر صفر رخ می‌دهد و کادر پیامی به شکل زیر ظاهر می‌شود :

شکل ۱۳ــ۳

با انتخاب Abort اجرای برنامه خاتمه می‌یابد. با انتخاب Retry مقدار i غیر صفر می‌شود و دوباره خط ۴ اجرا می‌شود که سبب رخ دادن خطا شده است. با انتخاب Ignore از خطای فعلی چشم‌پوشی شده و کنترل برنامه به خط پنجم منتقل می‌شود. به این ترتیب در زمان اجرا، خطا رفع می‌شود.

در مثال فوق با مفاهیم جدیدی روبه‌رو شدیم :

۱ــ شیء Err : از این شیء برای مدیریت خطاهای زمان اجرا استفاده می‌شود و شامل مشخصه‌هایی است که مهم‌ترین آن‌ها مشخصهٔ Number و Description است. مشخصهٔ Number شمارهٔ خطا و مشخصهٔ Description شرح خطای رخ داده را نگهداری می‌کنند. مشخصهٔ Number عددی Long و مشخصهٔ Description رشته‌ای است.

۲ــ دستور ON error : برای رفع خطاهای زمان اجرا از این دستور استفاده می‌شود. شکل کلی آن به صورت زیر است :

دستور On Error

هرگاه بخواهیم در صورت بروز خطا به محل خاصی از کد رجوع شود، می‌نویسیم :

برچسب On Error Goto

که این برچسب می‌تواند یک رشته یا عدد باشد، البته در صورت عدد بودن مقادیر منفی را نمی‌پذیرد.

اگر بخواهیم در صورت بروز خطا از آن صرف‌نظر کرده و به دستور بعدی رجوع شود، می‌نویسیم : Resume Next و هرگاه بخواهیم در صورت بروز خطا همان خط خطا را دوباره بخواند، می‌نویسیم : Resume

خودآزمایی

۱ــ برنامه‌ای بنویسید که لیستی از انواع خطاها را، که غالباً رخ می‌دهند، ارایه کند.

۲ــ توضیح دهید که اگر روالی یا تابعی فاقد قسمت رسیدگی به خطا باشد و در آن روال یا تابع خطایی به‌وجود آید، چه اتفاقی رخ می‌دهد.

۳ــ چرا رسیدگی به خطاها قبل از عبارت‌های Exit Sub یا Exit Function قرار می‌گیرند؟