فصل چهارم

برنامه‌نویسی شیءگرا
Object Oriented Programming (OOP)

هدف‌های رفتاری: فراگیر پس از پایان این فصل، خواهد توانست :

۱ــ مفاهیم کلاس، سلسله مراتب کلاس‌ها، وراثت، پنهان کردن اطلاعات و اعضای یک کلاس را شرح دهد.

۲ــ اصول ایجاد کلاس را شرح داده و کلاس جدیدی را ایجاد کند.

۳ــ با Object Browser کار کند.

۴ــ کنترل‌های ActiveX ایجاد کند.

۵ ــ با امکانات FSO آشنا شود و بتواند به کمک آن، با فایل‌ها، پوشه‌ها و درایوها کار کند و فایل‌ها را بخواند و بنویسد.

۶ ــ با متدهای مربوط به FSO کار کند و در برنامه‌های خود آن‌ها را به‌کار گیرد.

۱ــ۴ــ ایجاد شیءها از کلاس‌ها

نکته: در برنامه‌نویسی شیءگرا یک عنصر داده‌ای را مشخصه و یک روال را متد می‌نامند. معادل این واژه‌ها به ترتیب صفت و رفتار نیز است.

کلاس شبیه یک نوع داده‌ی تعریف شده به وسیله‌ی کاربر است با این تفاوت که علاوه بر داشتن داده‌ها، یک کلاس دارای روال‌هایی نیز است. کلاس مجموعه‌ای از مشخصه‌ها و روال‌هاست. یک شیء نمونه‌ای از یک کلاس است. به عنوان مثال، نوع داده‌ی Music که به صورت زیر تعریف شده است :

```
Type Music
    Composer As String
    Piece As String
End type
```

شامل روالی به نام Report است :

```
Sub Report ( )
    Dim Msg$
    Msg$ = = “Composer: “& Composer & vbCrLF
    Msg$ = Msg$ & “piece:” & piece
    ‘Display the string
    MsgBox Msg$
End Sub
```

اکنون فرض کنید که بتوانید روالی را به عنصر نوع داده‌ی Music اضافه کنید :

```
Type Music
    Composer As String
    Piece As String
    Report ( )
End Type
```

نمونه‌ای از این نوع داده را به صورت زیر تعریف می‌کنیم :

```
Dim MyMusic As Music
```

سپس مقادیری را برای آن تعیین می‌کنیم :

```
MyMusic. Composer = “Prince”
MyMusic.Piece = “Nothing Compares 2u”
```

بنابراین برای نمایش مقادیر عناصر، می‌توانید به صورت زیر عمل کنید :

```
MyMusic.Report()
```

این مثال، کل مطلب تئوری کلاس است. با این وجود، این مثال نشان می‌دهد که Scope (میدان دید) نقش بزرگ‌تری را ایفا می‌کند و ویژوال بیسیک انتظار دارد که مشخصه‌ها و متدهای

کلاس را برنامه‌نویس ایجاد کند.

۱ــ۱ــ۴ــ ایجاد کلاس در ویژوال بیسیک

کلاس شبیه کنترل ActiveX بدون رابط گرافیکی کاربر است. همان‌طور که بیان شد، می‌توان شیءهایی را از کلاس ایجاد کرد. بلاک ساخت کلاس، مدول کلاس است.

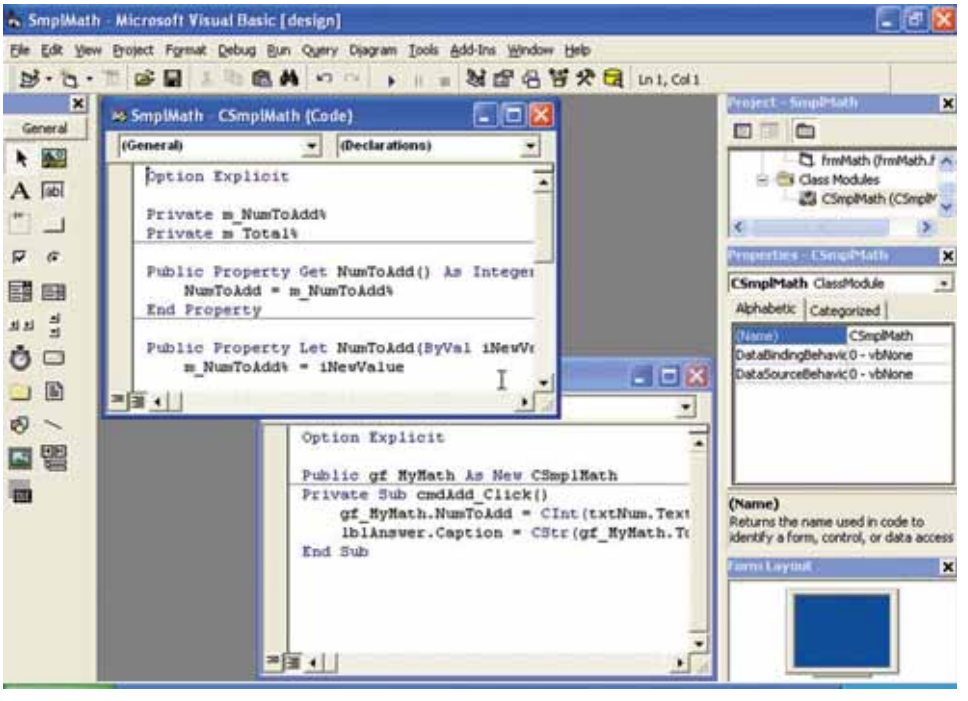
مثال ۱ــ۴:

۱ــ پروژه‌ی جدیدی را ایجاد کرده و نام آن را SmplMath.vbp قرار دهید. فرم پیش فرض را به frmMain تغییر دهید. مقدار مشخصه‌ی Caption فرم را به Simple Math Class تغییر دهید.

۲ــ از منوی Project گزینه‌ی Add Class Module را انتخاب کنید.

۳ــ پروژه‌ی جدید را برای شروع، انتخاب کنید.

۴ــ Name مدول کلاس را CSmplMath قرار دهید. مطمئن باشید که پنجره‌ی Properties باز است و سپس کلاس را در فایلی به نام CSmplMath ذخیره کنید (شکل ۱ــ۴).



شکل ۱ــ۴ــ مدول‌های کلاس به‌طور جداگانه در Project Explorer لیست می‌شوند.

۵ــ یک کنترل کادر متن، یک دکمه‌ی فرمان و یک برچسب به فرم frmMain اضافه کنید. مشخصه‌های Name و Caption کنترل‌ها را به‌صورت جدول ۱ــ۴ و اندازه و محل آن‌ها را به دلخواه تعیین کنید (شکل ۲ــ۴).

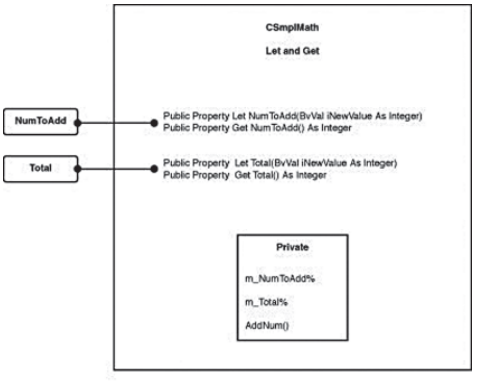
جدول ۱ــ۴ــ مشخصه‌های Name و Caption کنترل ها

کنترل	نام	مشخصه Text و Caption
TextBox	txtNum	خالی
CommandButton	cmdAdd	Add
Label	lblAnswer	خالی

شکل ۲ــ۴ــ داده‌های ارسال شده به شیء CsmplMath را از طریق فرم وارد کنید.

۲ــ۱ــ۴ــ اضافه کردن مشخصه‌ها به یک کلاس

در مثال فوق، ساختار پروژه شامل یک فرم و یک مدول کلاس بود. هم‌چنین نمونه‌ای از کلاس را ایجاد کردید که به وسیله‌ی مدول کلاس نمایش داده می‌شود. اکنون نیاز به ایجاد مشخصه‌هایی برای کلاس دارید. کلاس CSmplMath دارای دو مشخصه به نام‌های NumToAdd و Total است که هر دو از نوع Integer هستند. شکل ۳ــ۴، دیاگرام کلاس CSmplMath و مشخصه‌های آن را نمایش می‌دهد. همچنین روال‌های داخلی Let و Get را برای کلاس نشان می‌دهد که در ادامه شرح داده می‌شوند.



شکل ۳ــ۴ــ میدان دید داده‌ها و روال‌ها هنگام ایجاد کلاس، قطعی می‌شود. تنها داده‌هایی که بیرون از کلاس قابل رؤیت خواهند بود، مشخصه‌های کلاس هستند.

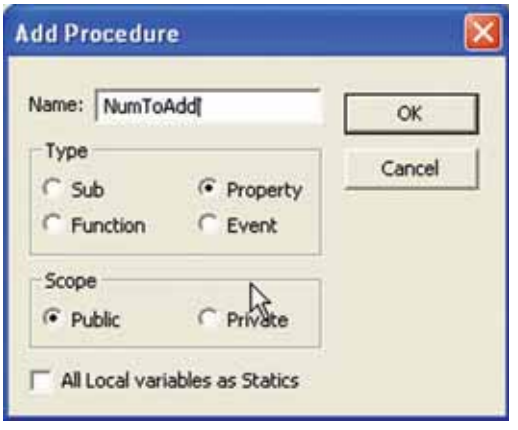
مثال ۲ــ۴ــ اضافه کردن مشخصه‌ها به کلاس مثال ۱ــ۴:

۱ــ مدول کلاس را در پنجره‌ی Code انتخاب کنید.

۲ــ از منوی Tools گزینه‌ی Add Procedure را انتخاب کنید.

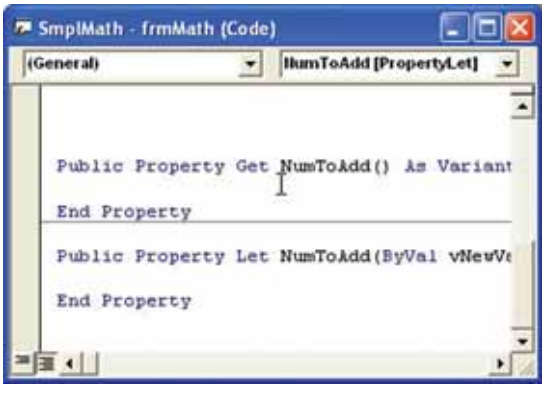
۳ــ گزینه‌ی Property را در کادر محاوره‌ای Add Procedure انتخاب کنید.

۴ــ مشخصه‌ی NumToAdd را در کادر متن Name اضافه کنید (شکل ۴ــ۴) روی OK کلیک کنید.



شکل ۴ــ۴ــ هنگامی که از کادر محاوره‌ای Add Procedure برای اضافه کردن مشخصه‌ای به کلاس استفاده می‌کنید، IDE ویژوال بیسیک، روال‌های Let و Get را برای آن مشخصه ایجاد می‌کند.

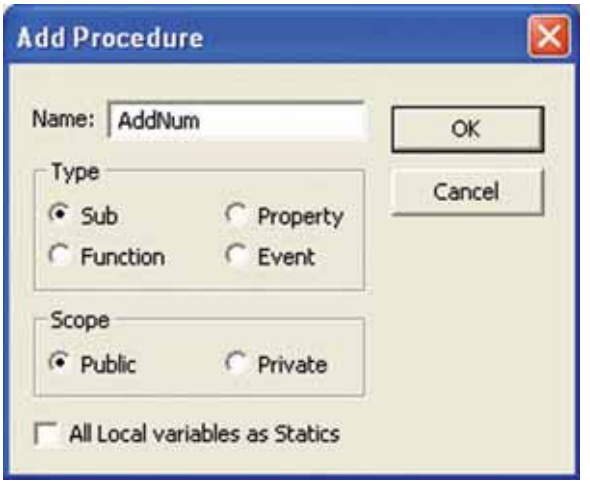
توجه کنید که کادر محاوره‌ای Add Procedure متدهای GetNumToAdd و LetNumToAdd را به پنجره‌ی Code اضافه می‌کند (شکل ۵ــ۴).



شکل ۵ــ۴ــ مقداردهی و دسترسی به داده‌های داخلی کلاس برای یک مشخصه را می‌توان به ترتیب با استفاده از روال‌های Let و Get انجام داد.

۵ــ مشخصه‌ی Total را از طریق کادر محاوره‌ای Add Procedure به مدول کلاس اضافه کنید.

۶ــ روال AddNum را با استفاده از کادر محاوره‌ای Add Procedure به مدول کلاس اضافه کنید (شکل ۶ــ۴). مطمئن باشید که گزینه‌ی Sub انتخاب شده باشد.



شکل ۶ــ۴ــ اگر میدان دید یک Sub را Public تعیین کنید، متدی از کلاس خواهد بود و در صورتی که Private باشد، فقط اعضای کلاس می‌توانند از آن Sub استفاده کنند.

۷ــ کد زیر را به بخش General مدول کلاس، اضافه کنید :

```
Private As Integer m_NumToAdd%
Private As Integer m_Total%
```

۸ــ روال‌های Get و Let مدول کلاس را مانند روال AddNum و به صورت زیر کامل کنید.

```
01 Public Property Get NumToAdd() As Integer
02 NumToAdd = m_NumToAdd%
03 End Property
04
05 Public Property Let NumToAdd (ByVal iNewValue As Integer)
06 m_NumToAdd% = iNewValue
07 AddNumbers
08 End Property
```

```
09
10 Public Property GetTotal() As Integer
11 Total = m_Total%
12 End Property
13
14 Public Property LetTotal (ByVal iNew Value As Integer)
15 m_Total% = iNewValue
16 End Property
17
18 Private Sub AddNumbers()
19 m_Total% = m_NumToAdd%+m_Total%
20 End Sub
```

۹ــ به روال‌های Get و Let کلاس رجوع کرده و نوع داده‌های Variant را به Integer تغییر دهید.

۱۰ــ کد را ذخیره کنید.

کلاس CSmplMath دارای دو مشخصه‌ی NumToAdd و Total است. هنگامی که مقداری را برای NumToAdd تعیین می‌کنید، از طریق کلاس به کل مجموعه اعمال می‌شود. سپس به‌طور داخلی این مقدار به مشخصه‌ی Total منتقل می‌شود.

در یک کلاس، داده‌ها و روال‌های Public و داده‌ها و روال‌های Private دارید. داده‌ها و روال‌های Public را مشخصه‌ها و متدها می‌نامند. داده‌ها و روال‌های Private را **متغیرهای عضو** و **توابع عضو** می‌نامند. این شمای Public/Private را کپسوله‌سازی (encapsulation) می‌گویند که هدف اصلی برنامه‌نویسی شیءگرا (Programming-OOP Object Oriented) است.

> **نکته:** روال‌های Get و Let خاص زبان ویژوال بیسیک هستند. سایر زبان‌های برنامه‌نویسی شیءگرا از روش‌های دیگر برای دسترسی به متغیرها و توابع عضو استفاده می‌کنند.

در برنامه‌نویسی شیءگرا برنامه‌ای که از کلاس استفاده می‌کند به‌طور مستقیم با متغیرها و توابع عضو کار نمی‌کند. در عوض، برنامه از متدهای کلاس می‌خواهد که به داده‌ها دسترسی داشته باشند. روال‌های دسترسی در ویژوال بیسیک، توابع Get و Let هستند. توجه داشته باشید که ویژوال بیسیک به‌طور خودکار روال‌های مشخصه را مدیریت می‌کند.

همان‌طور که مشاهده کردید، هنگام ایجاد مشخصه‌ی NumToAdd ، روال‌های Get و Let مربوط به مشخصه نیز ایجاد شدند. درون کلاس، دو متغیر عضو m_NumToAdd و m_Total در بخش General تعریف شدند. این متغیرها داده‌های واقعی کلاس را نگه می‌دارند. در خط ۶ کد فوق، روال Let مقدار زمان اجرای مشخصه‌ی NumToAdd را به متغیر عضو m_NumToAddf ارسال می‌کند. مقدار زمان اجرا با روال ()NumToAdd و به صورت آرگومان iNumValue ارسال می‌شود :

```
05 Public Property Let NumToAdd (ByVal iNewValue As Integer)
06 m_NumToAdd% = iNewValue
07 AddNumbers
08 End Property
```

نکته: به‌طور متعارف، متغیرهای عضو با پیشوند m مشخص می‌شوند.

روال ()Get NumToAdd که به صورت یک تابع است، مشخصه‌ی NumToAdd را در خط ۲ با مقدار m_NumToAdd مقداردهی می‌کند.

در خطوط ۱۰ تا ۱۶ نیز همین عملیات برای مشخصه‌ی Total انجام می‌شود.

روال ()AddNum یک روال محلی برای کلاس است. این روال متغیر عضو m_NumToAdd را با m_Total جمع می‌کند.

نکته: هنگامی که مقداری را برای مشخصه‌ی یک شیء قرار می‌دهید، روال Let آن مشخصه و هنگام به‌دست آوردن مقدار یک مشخصه، روال Get آن مشخصه فراخوانی می‌شوند.

۳ــ۱ــ۴ــ ایجاد شیء از یک کلاس

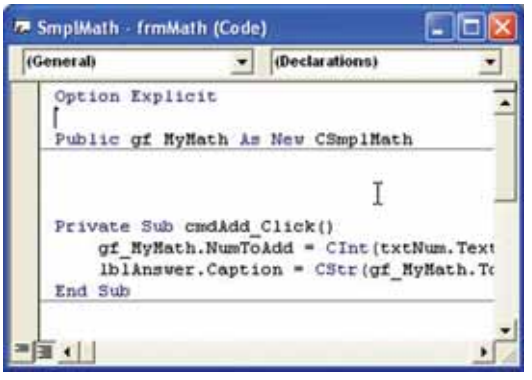
بعد از ایجاد کلاس، می‌توان نمونه‌ای از آن را به عنوان شیء در برنامه به‌کار برد.

مثال ۳ــ۴ــ ایجاد یک شیء براساس کلاس CSmplMath

۱ــ فرم frmMain را از Project Explorer انتخاب کرده و روی دکمه‌ی View Code کلیک کنید.

۲ــ کد زیر را به بخش General فرم اضافه کنید (شکل ۷ــ۴).

```
Public gf_MyMath As New CSmplMath
```



شکل ۷ــ۴ــ در صورتی که می‌خواهید یک شیء در طول اجرای برنامه پایدار باشد، باید آن را در بخش General ایجاد کنید.

۳ــ کد زیر را به روال رویداد cmdAdd_click() اضافه کنید.

```
gf_MyMath.NumToAdd = Cint(txtNum.Text)
lblAnswer.Caption= cstr(gf_MyMath.Total)
```

۴ــ کد را ذخیره و اجرا کنید.

اولین شیء را هنگامی که متغیر gf_MyMath را در بخش General فرم اعلان کردید، ایجاد نموده‌اید. شکل کلی ایجاد یک شیء به صورت زیر است :

```
Dim | Public | Private MyObject As New MyClass
```

در این دستور :

Dim | Public | Private کلید واژه‌های تعریف میدان دید شیء هستند.

- MyObject نام شیء است.
- As کلید واژه‌ی تعیین تعریف نوع است.

- New کلید واژه‌ای است که یک نمونه از شیء را ایجاد می‌کند.
- MyClass کلاسی است که شیء نمونه‌ای از آن خواهد بود.

بنابراین، دستور Public gf_MyMath As New CSmplMath یک شیء ایجاد می‌کند که می‌تواند براساس کلاس در کد مورد استفاده قرار گیرد.

در صورتی که می‌خواهید از این کلاس در پروژه‌های دیگری استفاده کنید، نیاز دارید که فایل مدول کلاس (CSmplMath.cls) را به پروژه‌ی جدید اضافه کنید و سپس شیئی را در داخل فرم با مدول پروژه با استفاده از کلید واژه‌ی New ایجاد کنید.

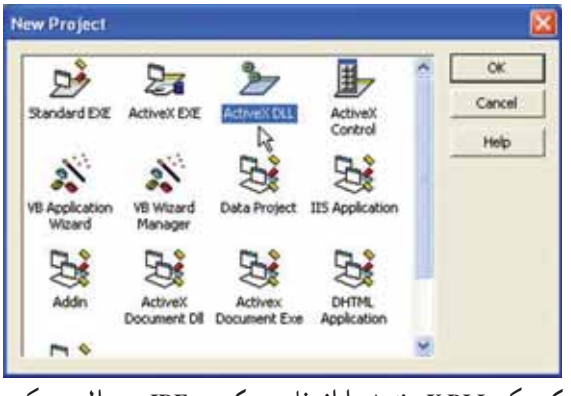
۲ــ۴ــ ایجاد ActiveX DLL

اگر فایل cls یکی از فایل‌های پروژه باشد، کلاسی که ایجاد کرده‌اید فقط می‌تواند داخل همان پروژه مورد استفاده قرار گیرد. در صورتی که می‌خواهید این کلاس برای سایر برنامه‌ها در زمان اجرا به عنوان یک DLL قابل دسترس باشد، می‌توانید این کار را با یک کلاس ActiveX DLL انجام دهید.

نکته: ActiveX DLL ها مستقل از زبان هستند. آن‌ها را می‌توان در هر زبانی که از Component Object Model (COM) پشتیبانی می‌کند، نوشت.

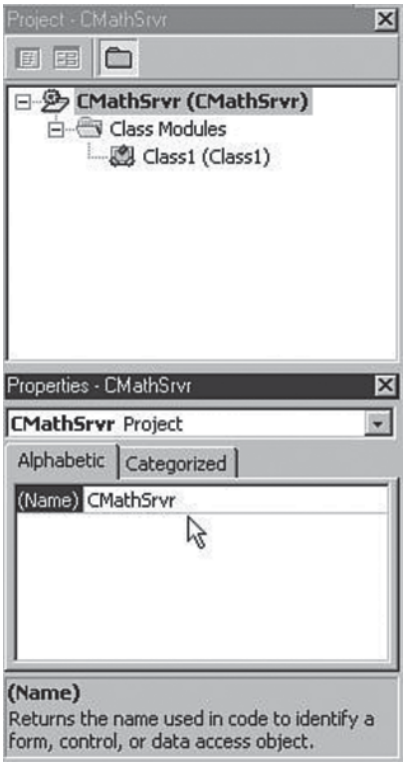
مثال ۴ــ۴ــ ایجاد یک ActiveX DLL با استفاده از کلاس CSmplMath :

۱ــ پروژه‌ی جدیدی را ایجاد کنید. ActiveX DLL را به عنوان نوع پروژه انتخاب کنید (شکل ۸ ــ ۴).



شکل ۸ــ۴ــ هنگامی که یک ActiveX DLL را انتخاب می‌کنید، IDE ویژوال بیسیک به‌طور خودکار پروژه را تنظیم می‌کند.

۲ــ پروژه را در Project Explorer انتخاب کرده و نام آن را Cmathsrvr قرار دهید (شکل ۹ــ۴). نام فایل پروژه را نیز همین نام انتخاب کنید.



شکل ۹ــ۴ــ نام پیش‌فرض ActiveX DLL نام پروژه خواهد بود و نه نام کلاس

۳ــ در پنجره‌ی Project Explorer کلیک راست کرده و Add را از منوی میانبر انتخاب کرده و سپس Add File را انتخاب کنید.

نکته: هنگامی که یک ActiveX DLL را ایجاد می‌کنید، یک فایل کلاس پیش‌فرض ارایه می‌کنید. در صورتی که می‌خواهید کلاس را به عنوان بخشی از پروژه ایجاد کنید، باید فایل کلاس را اضافه کنید.

۴ــ فایل CSmplMath.cls را برای کلاس اضافه کنید.

۵ــ روی فایل کلاس پیش فرض (Classl.cls) در Project Explorer کلیک راست کرده و

گزینه‌ی Remove Classl.cls را انتخاب کنید.

۶ــ از منوی Project ، گزینه‌ی CMathSrvr Properties را انتخاب کنید.

۷ــ در کادر متن Project Description عبارت Simple ActiveX DLL Demo را وارد کنید (شکل ۱۰ــ۴).

شکل ۱۰ــ۴ــ کادر متن Project Description سبب نمایش توضیحی درباره‌ی کلاس در Object Browser و کادر محاوره‌ای Tools/References می‌شود.

۸ــ کلاس CSmplMath را در Project Explorer انتخاب کنید. در پنجره‌ی Properties، مشخصه‌ی Instancing را با 5-MultiUse مقداردهی کنید (شکل ۱۱ــ۴).

شکل ۱۱ــ۴ــ می‌توان چندین نوع مختلف از Automation servers داشت. MultiUse بدین معنی است که تمام شیءهای ایجاد شده از این کلاس یک نمونه از سرور را به اشتراک خواهند گذاشت.

عملیات کامپایل ActiveX DLL شبیه پروژه‌هایی اجرایی است. برای ارایه نسخه‌ی نصبی این نوع پروژه نیز از Setup Wizard استفاده کنید تا تمام فایل‌های مورد نیاز به همراه آن ارایه شوند.

۹ــ از منوی File گزینه‌ی Make Cmathsrvr.dll را انتخاب کنید.

قبل از این که بتوانید از کلاس استفاده کنید، باید DLL را به IDE ویژوال بیسیک اضافه کنید.

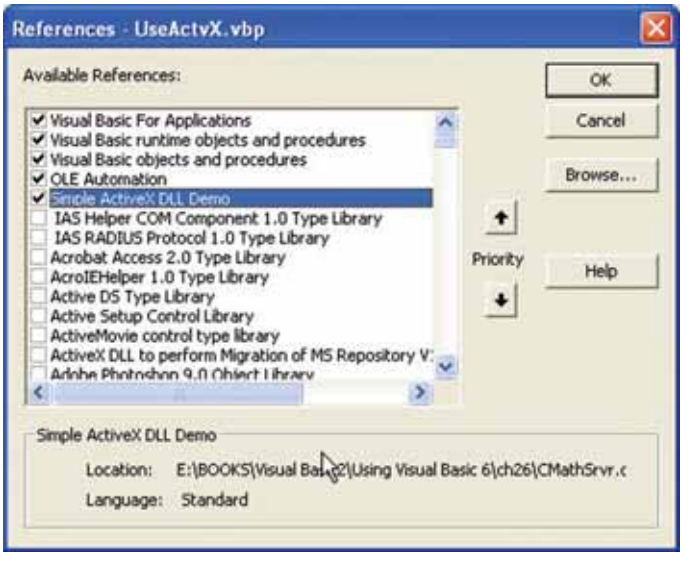
مثال ۵ــ۴ــ اضافه کردن یک ActiveX DLL به کد ویژوال بیسیک

۱ــ پروژه‌ی جدیدی را ایجاد کرده و نام آن را UserActvX قرار دهید. نام فرم پیش‌فرض را frmMain و مقدار مشخصه‌ی Caption فرم را Using a ActiveX DLL قرار دهید.

۲ــ یک کنترل کادر متن، دکمه‌ی فرمان و برچسب به فرم اضافه کنید. مشخصه های Name و Caption را مطابق جدول ۱ــ۴ تنظیم کنید. کنترل‌ها را مطابق شکل ۲ــ۴ روی فرم قرار دهید.

۳ــ از منوی Project گزینه‌ی References را انتخاب کنید.

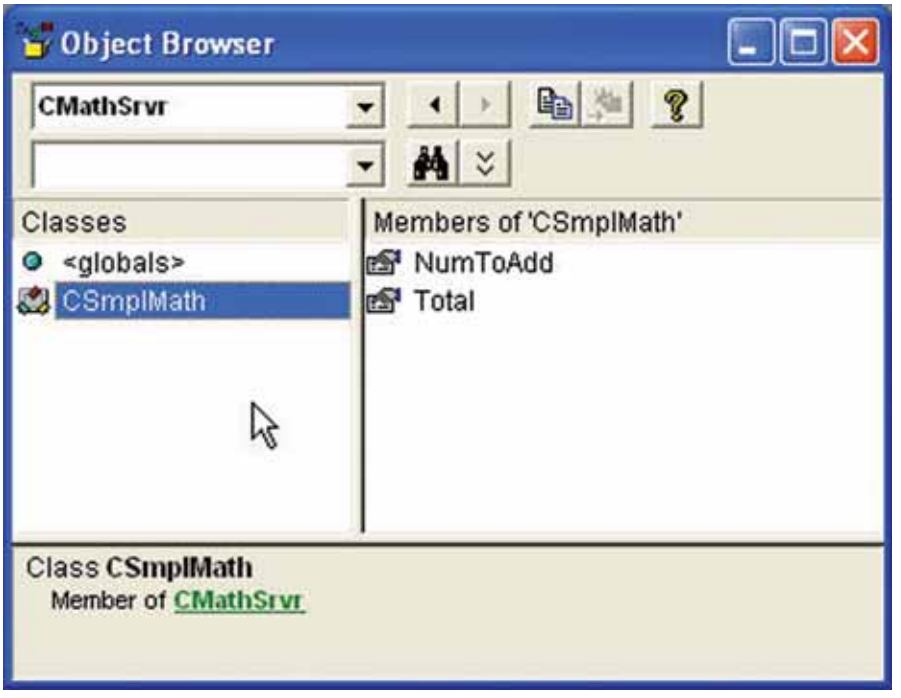
۴ــ مطمئن شوید که ActiveX DLL در کادر محاوره‌ای Reference با شرح آن، لیست شده است (شکل ۱۲ــ۴). در صورتی که نبود، روی دکمه‌ی Browse کلیک کنید تا کادر محاوره‌ای Add Refernce باز شود. فایل CMathSrvr.Dll را انتخاب کنید.



شکل ۱۲ــ۴ــ اگر ActiveX DLL با سیستم‌تان رجیستر شده است. کلاس‌های DLL در کادر محاوره‌ای References ظاهر خواهند شد. در غیر این‌صورت، نیاز دارید که DLL را به‌طور مستقیم با استفاده از دکمه‌ی Browse انتخاب کنید.

۵ــ برای ایجاد کلاس‌های جدیدی در ActiveX DLL که برای پروژه قابل دسترس باشند، کادر علامت CMathSrvr را انتخاب کنید. سپس روی OK کلیک کنید.

۶ــ کلید F2 را فشار دهید تا Object Browser نمایش داده شود. کتابخانه‌ی کلاس CmathSrvr را از لیست باز شو در گوشه‌ی سمت چپ بالا انتخاب کنید (شکل ۱۳ــ۴).



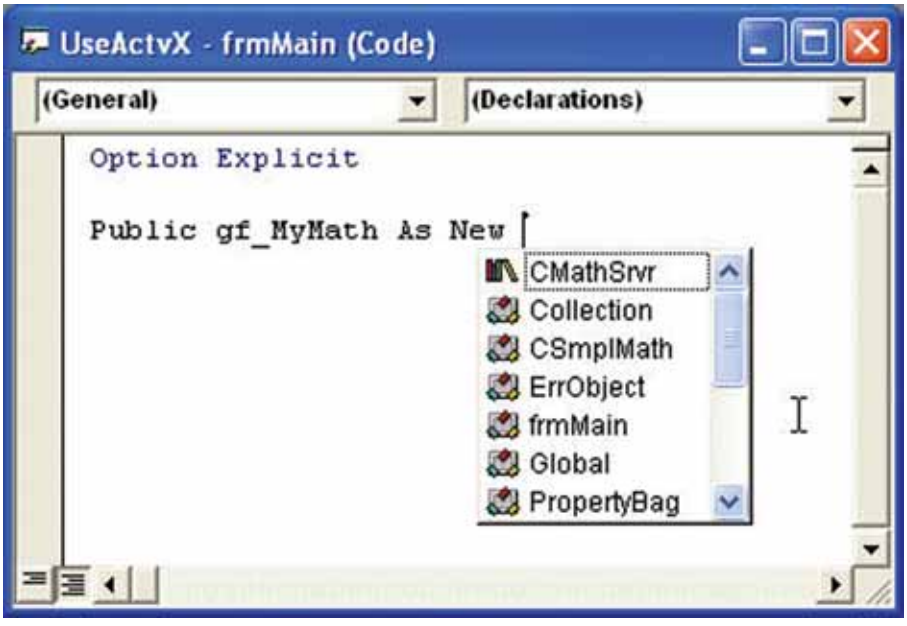
شکل ۱۳ــ۴ــ Object Browser امکان جستجوی تمام مشخصه‌های کلاس و متدهای درون یک مؤلفه‌ی ActiveX را فراهم می‌کند.

۷ــ کلاس CSmplMath را از لیست کلاس‌ها انتخاب کنید. روی آیکون Copy در بالای Object Browser کلیک کنید.

۸ــ کد زیر را در بخش General فرم frmMain وارد کنید :

```
Public gf_MyMain As New
```

۹ــ نام کلاس را از حافظه‌ی کلیپ برد، Paste کنید تا کد فوق کامل شود. در صورتی که نمی‌خواهید شیء را از Object Browser کپی کنید، می‌توانید شیء CSmplMath را از لیست بازشوی auto = complete انتخاب کنید (شکل ۱۴ــ۴).



شکل ۱۴ــ۴ــ هنگامی که کلاس را به پروژه اضافه می‌کنید، این کلاس به‌طور خودکار در لیست بازشوی /Methods Objects از IDE ویژوال بیسیک ظاهر می‌شود.

۱۰ــ کد زیر را به روال رویداد cmdAdd_click() اضافه کنید :

```
gf_MyMath.NumToAdd = cInt(txtNum.Text)
lblAnswer.Caption = Cstr(gf_MyMath.Total)
```

۱۱ــ کد را ذخیره کرده و اجرا کنید (شکل ۱۵ــ۴).

شکل ۱۵ــ۴ــ این برنامه ممکن است مشابه برنامه‌ی قبلی باشد ولی این‌طور نیست. این کد کلاسی را فراخوانی می‌کند که بخشی از یک DLL خارجی است. در حالی که کد قبلی داخل فایل اجرایی کامپایل می‌شود.

مثال ۶ــ۴ــ کار کردن با مؤلفه‌ها: پروژه‌ای به نام MthFuncs.vbp ایجاد کنید که منبع مؤلفه‌ی ActiveX به نام SimpleMathFunctions است. مؤلفه‌ی ActiveX مفهومی برای یک ActiveX DLL است که شامل یک یا چند کلاس است.

مؤلفه‌ی SimpleMathFunctions یک بهبود در کاری است که با CSmplMath انجام

می‌دادید. این مؤلفه شامل کلاس CMathFunc است که دارای سه مشخصه‌ی NewValue، Operation و TotalValue است. همچنین متدی به نام DoOperation دارد.

روشی که کلاس کار می‌کند، تعیین یک مقدار عددی برای مشخصه‌ی NewValue و سپس مقداری برای مشخصه‌ی Operation است. Operation می‌تواند یکی از چهار مقدار زیر را داشته باشد :

1. Addition
2. Subtrction
3. Multiplication
4. Division

بعد از این که مشخصه‌های NewValue و Operation با مقادیر مناسبی مقداردهی شدند، متد DoOperation فراخوانی می‌شود. نتیجه در مشخصه‌ی فقط خواندنی TotalValue نمایش داده خواهد شد.

این مؤلفه‌ی ActiveX یک شرح کاملی از ActiveX DLLها است. به خاطر داشته باشید، هنگامی که از این کد استفاده می‌کنید، مطمئن شوید که DLL روی سیستم رجیستر شده یا در IDE ویژوال بیسیک وجود دارد. از کادر محاوره‌ای Add Reference برای در دسترس قرار دادن مؤلفه برای برنامه استفاده کنید.

ActiveX DLL ها یک بخش کلیدی از Microsoft Distributed Computing Technology هستند. نه تنها می‌توان این مؤلفه‌ها را برای استفاده در کامپیوترهای شخصی طراحی کرد بلکه می‌توان آن‌ها را روی کامپیوترهای راه دور نیز قابل دسترس کرد. حتی می‌توان این ActiveX DLL ها را به عنوان مؤلفه‌ها یا برنامه‌های کاربردی روی Microsoft's Internet Information Server (IIS) اجرا کرد.

۳ــ۴ــ ایجاد کنترل‌های ActiveX

هر زمانی که نیاز به کنترل دارید (علی‌الخصوص در یک محیط کاری)، همیشه باید بررسی کنید که آیا کنترل موجود است یا نه؟ در صورت نبودن کنترل، آن را خودتان ایجاد کنید. ایجاد کنترل‌ها زمان و هزینه‌ی زیادی را دربرخواهد داشت.

با ویژوال بیسیک می‌توان کنترل‌های ActiveX ایجاد کرد که نه تنها در برنامه‌نویسی ویژوال

بیسیک، بلکه داخل سایر محیط‌های برنامه‌نویسی مثل C++ و Delphi نیز می‌توانند مورد استفاده قرار گیرند. اگر برای محیط اینترنت، برنامه می‌نویسید، کنترل‌های ActiveX وظایف را در سطح جدیدی انجام داده و با صفحات وب محاوره می‌کنند.

مثال ۷ــ۴ــ ایجاد کنترل ActiveX

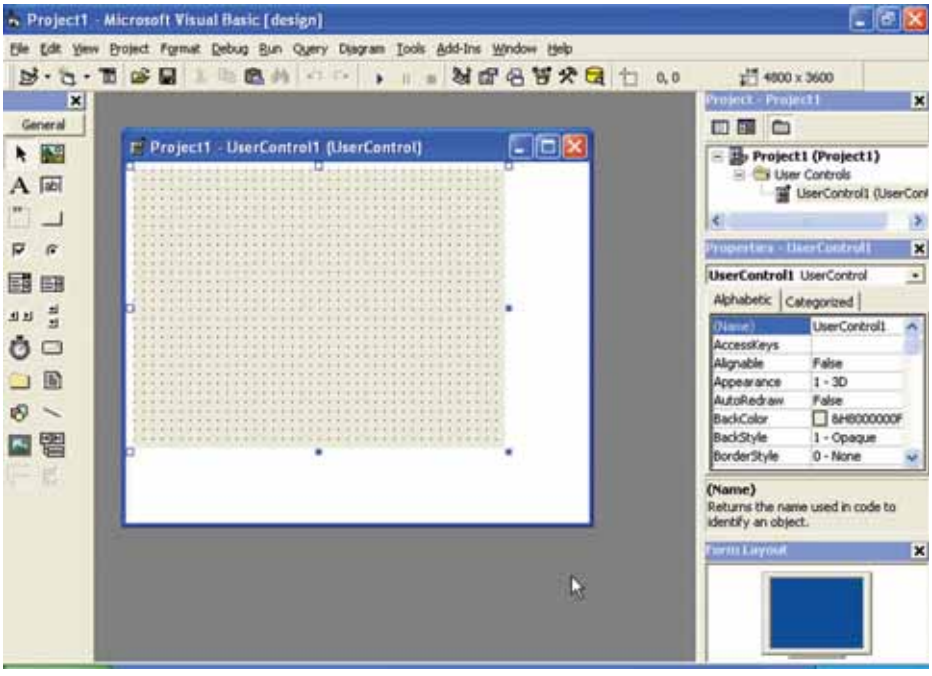
۱ــ پروژه‌ی جدیدی را شروع کنید. در کادر محاوره‌ای New Project گزینه‌ی ActiveX Control را انتخاب کنید.

شکل ۱۶ــ۴ــ کنترل ActiveX را داخل یک پروژه‌ی جداگانه ایجاد کنید.

نکته: UserControl نام صفحه‌ای است که ویژوال بیسیک برای ایجاد یک کنترل سفارشی، کنترل‌های ذاتی را به آن اعمال می‌کند. اگر از کنترل‌های غیرذاتی استفاده می‌کنید، مطمئن باشید که آن‌ها را به‌طور ایمن به‌کار می‌برید.

۲ــ پروژه را CursorReporter نامگذاری کرده و در فایل rptcursr.vbp ذخیره کنید.

۳ــ نام UserControl را از UserControll به CursorReport تغییر دهید (شکل ۱۷ــ۴).



شکل ۱۷ــ۴ــ User Control اساس شیئی است که دارای چهار عنصر است: User Control، Designer، Properties و ToolBoxPicture.

۴ــ یک کنترل برچسب به UserControl Designer اضافه کنید. مشخصه‌های کنترل را با مقادیری مثل جدول ۲ــ۴، تنظیم کنید.

جدول ۲ــ۴ــ تنظیمات مشخصه‌های کنترل برچسب

Property	value
Name	lblMain
BorderStyle	1-Fixed Single
BackColor	&H0000FFFF
Caption	(blank)
Height	375
Left	0
Top	0
Width	1215

۵ــ مقدار ToolBoxBitmap را در پنجره‌ی Properties انتخاب کنید. روی دکمه‌ی ... کلیک کرده و تصویر طرح بیتی Cursor.bmp را برای مشخصه‌ی ToolBoxPicture تعیین کنید (شکل ۱۸ــ۴).

شکل ۱۸ــ۴ــ ToolBoxBitmap تصویری است که کنترل در جعبه ابزار ویژوال بیسیک نمایش خواهد داد.

۶ــ در برگیرنده‌ی UserControl را تغییر اندازه دهید تا کمی بزرگتر از کنترل Label باشد (شکل ۱۹ــ۴).

شکل ۱۹ــ۴ــ هنگامی که کنترل ActiveX را به یک فرم برنامه‌ی کاربردی اضافه می‌کنید، ابعاد اولیه‌ی کنترل به اندازه‌ی User Control Designer خواهد بود.

۷ــ پروژه را ذخیره کنید. مطمئن باشید که فایل‌ها به صورت زیر نامگذاری شده‌اند :

Type	Name	File name
Project	Cursor Reporter	rpcursr.vbp
UserControl	CursorReport	rptcursr.ctl

۸ــ Close را از منوی کنترلی انتخاب کنید (آیکون روی نوار منو).

اکنون ToolBoxPicture کنترل ActiveX در جعبه ابزار ویژوال بیسیک ظاهر می‌شود (شکل ۲۰ــ۴).

شکل ۲۰ــ۴ــ کنترل CursorReport ActiveX در جعبه ابزار ویژوال بیسیک از فایل طرح بیتی مکان‌نمای ویندوز ایجاد شده است.

دقیقاً اصول ایجاد یک کنترل ActiveX را انجام دادید. کنترل را مانند سایر کنترل‌های ActiveX به فرم اضافه کنید.

۴ــ۴ــ آشنایی با شیء User Control

در درون هر کنترل ActiveX که با ویژوال بیسیک ایجاد می‌شود. شیء UserControl

وجود دارد. شیء UserControl اساس چیزی است که می‌توانید به سایر کنترل‌های ActiveX اضافه کنید تا یک کنترل جدید منحصر به فردی را ایجاد کنید. کنترل‌هایی که به شیء UserControl اضافه می‌شوند را کنترل‌های Constituent (ترکیب‌کننده) می‌نامند. در مثال قبلی، این نوع کنترل، کنترل برچسب lblMain نام داشت.

یک شیء مهم هنگام ایجاد کنترل‌های ActiveX، شیء AmbientProperties است. این شیء تنظیم مشخصه‌های فرم یا شیء دربرگیرنده‌ی دیگری که کنترل ActiveX به کار می‌برد را به‌عهده دارد. شیء Ambient Properties با رجوع به مشخصه‌ی Ambient شیء UserControl قابل دسترس است. به عنوان مثال، در صورتی که می‌خواهید مقدار مشخصه‌ی BackColor مربوط به UserControl را با همان مقدار مشخصه‌ی BackColor دربرگیرنده‌ی ActiveX تنظیم کنید، از کد زیر استفاده کنید :

```
Usercontrol. Backcolor = Ambient. BackColor
```

۱ــ۴ــ۴ ــ اضافه کردن UserControl به فرم

مثال ۸ ــ۴ــ اضافه کردن کنترل CursorReport به فرم جدیدی از پروژه:

۱ــ به پروژه‌ی rptCursr.vbp که در مثال قبل ایجاد کردید، برگردید، مطمئن شوید که پنجره‌ی UserControl Designer بسته شده باشد و آیکون CursorReport در جعبه ابزار فعال شده است.

۲ــ از منوی File گزینه‌ی Add Project را انتخاب کنید (New Project را انتخاب نکنید).

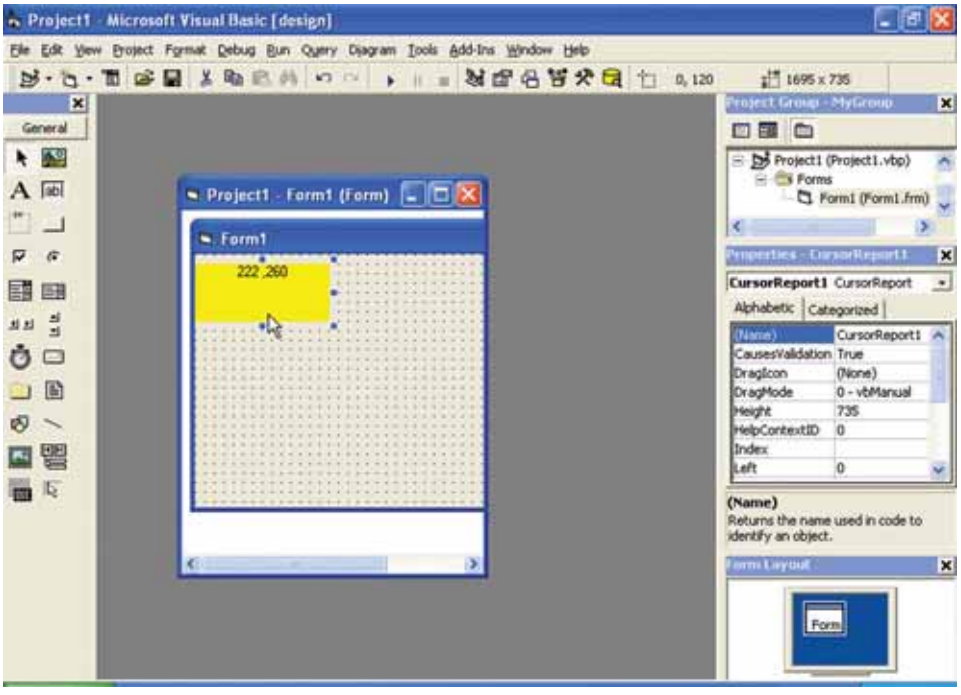
۳ــ از کادر محاوره‌ای Add Project، گزینه‌ی Standard EXE را انتخاب کنید.

۴ــ کنترل Cursor report را انتخاب کرده و به فرم اصلی در پروژه‌ی جدید اضافه کنید (شکل ۲۱ــ۴).

۵ــ از منوی File گزینه‌ی Save Project Group را انتخاب کنید. پروژه‌ی اضافه شده را با نام پیش‌فرض ذخیره کرده و گروه پروژه را در فایل MyGroup.vbg ذخیره کنید.

نکته: ویژوال بیسیک امکان کار کردن با چندین پروژه را به‌طور همزمان فراهم می‌کند. مجموعه‌ی پروژه‌ها به صورت یک گروه پروژه در فایل vbg. ذخیره می‌شوند. هم‌چنین می‌توان با چندین پروژه به‌طور مستقل از گروه پروژه کار کرد.

۶ــ برای اجرای پروژه‌ی جدید، کلید F5 را فشار دهید.



شکل ۲۱ــ۴ــ کنترل CursorReport را از جعبه ابزار انتخاب کنید.

۵ــ۴ــ اضافه کردن وظایف به کنترل ActiveX

> **نکته:** هنگامی که بیش از یک پروژه به IDE اضافه کنید، ویژوال بیسیک سعی خواهد کرد که گروه پروژه را برای همه‌ی پروژه‌ها ایجاد کند.

تا این‌جا یاد گرفتید که چگونه می‌توان کنترل ذاتی را به عنوان بخشی از کنترل ActiveX درنظر گرفت. اکنون می‌توانید وظایفی را به کنترل اضافه کنید تا برای سایر برنامه‌نویسان مفید باشد.

می‌خواهیم کنترلی که در قسمت قبلی ایجاد کرده‌ایم را بهینه‌سازی کنیم تا محل اشاره‌گر ماوس را در هر جایی از صفحه نمایش، گزارش کند. کنترل ActiveX قابلیت تغییر اندازه نیز خواهد داشت.

به‌طور عادی، برنامه‌هایی که در ویژوال بیسیک نوشته می‌شوند، نمی‌توانند محل اشاره‌گر ماوس را در هر جایی غیر از محدوده‌ی فرم‌هایی که بخشی از برنامه‌ی کاربردی هستند، گزارش کنند. برای این‌که کنترل امکان گزارش محل اشاره‌گر ماوس روی صفحه را داشته باشد، از API ویندوز

به نام GetCursorPos استفاده کنید.

برای این که کنترل ActiveX محل اشاره‌گر روی صفحه را در هر بار تعیین کند، یک کنترل Timer به UserControl اضافه کرده و روال رویداد Timer آن را برنامه‌نویسی کرده و تابع GetCursorPos را فراخوانی کنید. این تابع، محل اشاره‌گر روی صفحه را برمی‌گرداند. بعد از فراخوانی تابع، رشته‌ی برگردانده شده را در مشخصه‌ی Caption کنترل ترکیب‌کننده‌ی lblMain قراردهید.

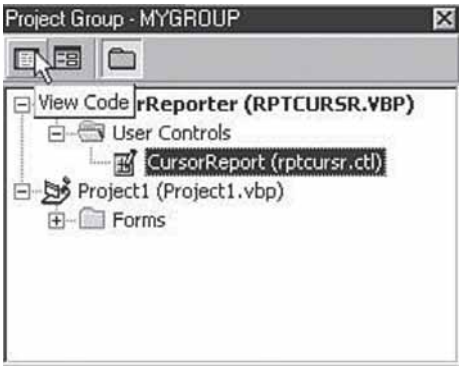
مثال ۹ــ۴ــ تنظیم کنترل ActiveX برای نمایش محل اشاره‌گر

۱ــ به گروه پروژه‌ی MyGroup.vbg که شامل کنترل CursorReport است، برگردید.

۲ــ کنترل Timer را به کنترل ActiveX اضافه کنید. مقدار مشخصه‌ی Interval را با ۱۰۰ مقداردهی کنید.

۳ــ کنترل ActiveX سفارشی را در پنجره‌ی Project Explorer انتخاب کنید (کنترل CursorReport که در فایل rptcursor.vbp ذخیره شده است).

۴ــ روی دکمه‌ی View Code کلیک کنید تا پنجره‌ی Code مربوط به کنترل ActiveX نمایش داده شود (شکل ۲۲ــ۴).



شکل ۲۲ــ۴ــ با کلیک کردن روی دکمه‌ی View Code می‌توان کد مربوط به هر کنترل یا مدول را مشاهده کرد.

۵ــ کد زیر را به بخش General Declarations کنترل ActiveX اضافه کنید :

```
Private Declare Function GetCursorPos_Lib "user32"
(IpPoint As POINTAPI) As Long
Private Type POINTAPI
```

```
    x As Long
    y As Long
    End Type
```

۶ــ کد زیر را به روال رویداد Timer1_Timer() اضافه کنید :

```
01 `Return Variable for the API function
02 `Dim L As Long
03
04 `Variable of the Windows defined type, POINTAPI.
05 `This is the structure to which the API function
06 `Will return the coordinates of the cursor
07 `Dim pt As POINTAPI
08
09 `Get the mouse pointer coordindates
10 l = GetCursorPos (pt)
11
12 `Get the x and y elements from the POINTAPI type
13 `and create a display string. Assign that string
14 `to the constituent label's Caption property.
15 lblMain.Caption = CStr(pt.x)&", "&CStr(pt.y)
```

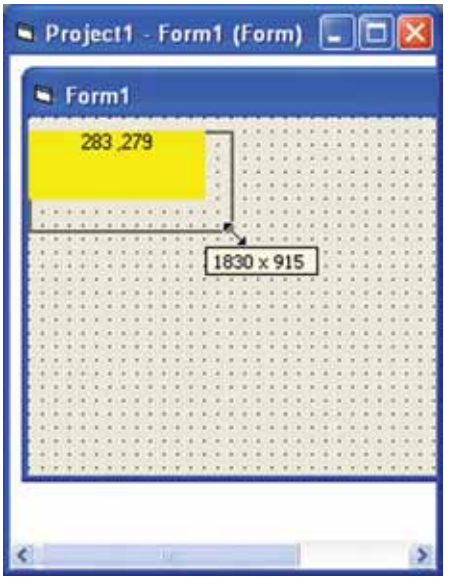
۷ــ کد زیر را به روال رویداد UserControl_Resize() اضافه کنید :

```
lblMain.Width = UserControl.Width
lblMain.Height = UserControl.Height
```

۸ــ پنجره‌ی UserControl را ببندید.

۹ــ پنجره‌ی Form Designer را برای فرم اصلی پروژه‌ی دیگری (پروژه‌ای که از کنترل ActiveX سفارشی استفاده می‌کند) در گروه پروژه باز کنید. توجه داشته باشید که اکنون کنترل ActiveX سفارشی، محل اشاره‌گر روی صفحه نمایش را گزارش می‌کند (شکل ۲۳ــ۴).

۱۰ــ تمام پروژه‌های داخل گروه پروژه و به همان ترتیب گروه پروژه را نیز ذخیره کنید.



شکل ۲۳ــ۴ــ تغییر اندازه‌های کنترل Label داخل روال رویداد Resize مربوط به UserControl .

اکنون یک کنترل ActiveX کاملی دارید که محل اشاره‌گر روی صفحه را نمایش می‌دهد. این کنترل را می‌توان به هر برنامه‌ای اضافه کرد (چه در ویژوال بیسیک و چه در Visual C++). همچنین می‌توان کنترل را به صفحه‌ی وب نیز اضافه کرد.

قبل از این که بتوانید از این کنترل در محیط‌های مختلف استفاده کنید، باید آن را به یک فایل OCX کامپایل کنید.

۱ــ۵ــ۴ــ کامپایل کردن کنترل‌های ActiveX سفارشی

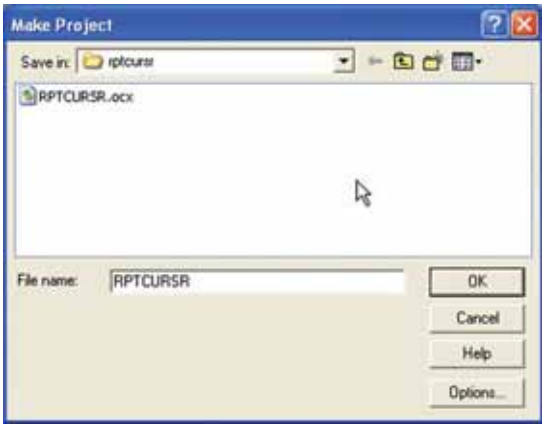
قبل از ارایه‌ی کنترل ActiveX سفارشی، باید آن را به فایل OCX کامپایل کرد. به‌طور پیش‌فرض، فایل OCX همان نام فایل پروژه را خواهد داشت. مراحل انجام این کار به صورت زیر است :

۱ــ پروژه‌ی CursorReporter را در Project Explorer انتخاب کنید.

۲ــ از منوی File گزینه‌ی Make RPTCURSR.ocx را انتخاب کنید تا کادر محاوره‌ای Make Project نمایش یابد.

۳ــ در کادر محاوره‌ای Make Project روی OK کلیک کنید (شکل ۲۴ــ۴).

۴ــ پروژه را ذخیره کرده و گروه پروژه را ببندید.



شکل ۲۴ــ۴ــ عملیات ایجاد یک OCX شبیه ایجاد یک فایل EXE است. اگر روی دکمه‌ی Options کلیک کنید، می‌توانید شماره نسخه و سایر جزئیات را تنظیم کنید.

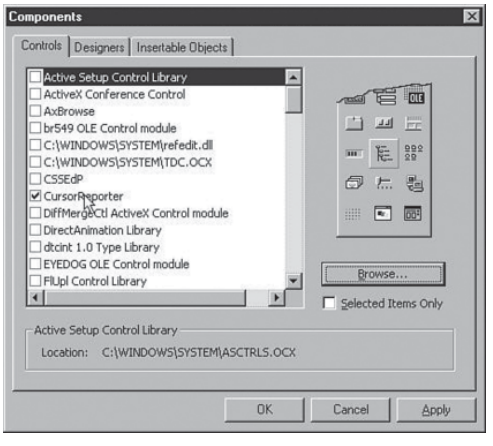
بعد از کامپایل کنترل ActiveX سفارشی با یک OCX نیاز به استفاده از تکنیک «دوپروژه‌ای» برای کار کردن با کنترل‌های ActiveX ندارید و به سادگی از کنترل ActiveX مانند هر کنترل دیگری استفاده کنید. می‌توانید از ویزارد Package and Deployment برای توزیع OCX به سایر برنامه‌نویسان استفاده کنید. مراحل زیر، چگونگی استفاده از کنترل در پروژه‌ها را نشان می‌دهند :

۱ــ از منوی Projects گزینه‌ی Components را انتخاب کنید.

۲ــ در کادر محاوره‌ای Components ، روی Browse کلیک کنید.

۳ــ در کادر محاوره‌ای Add ActiveX Control ، پوشه‌ای که شامل کنترل ActiveX است را پیدا کنید.

۴ــ کنترل را از لیست Components انتخاب کنید (شکل ۲۵ــ۴).



شکل ۲۵ــ۴ــ کنترل ActiveX سفارشی را مانند هر کنترل دیگری به IDE ویژوال بیسیک اضافه کنید.

بعد از کامپایل پروژه‌ی ActiveX به یک فایل OCX ، می‌توانید آن را ارایه کنید. برای ارایه‌ی OCX باید از Setup Wizard استفاده کنید. مانند سایر پروژه‌های ویژوال بیسیک ، فایل‌های اصلی مورد نیاز زمان اجرا باید با OCX همراه باشند.

هم‌چنین OCX باید روی سیستم کاربر، رجیستر شود. ویزارد Package and Deployment به‌طور خودکار همه‌ی این عملیات Setup را انجام می‌دهد. اگر کنترل ActiveX خاصی که از کنترل‌های دیگری غیر از کنترل‌های استاندارد ویژوال بیسیک استفاده می‌کند را ایجاد کنید، این کنترل‌ها نیز باید به همراه کنترل خاصی در ویزارد Package and Deployment ارایه شوند.

به‌طور عملی تمام برنامه‌های ویندوز باید به‌صورت یکسان نصب شوند. براساس معماری جدید، اطلاعات زیادی باید در رجیستری ویندوز ثبت شوند. این اطلاعات هنگام اجرای Setup.exe مربوط به برنامه یا کنترل ActiveX وارد می‌شوند.

داشتن نسخه‌ی Uninstall نیز برای یک برنامه لازم است. ویزارد Package and Deployment به‌طور خودکار این گزینه را ایجاد می‌کند.

اگر قصد دارید که کنترل ActiveX را در صفحه‌ی وب یا اینترنت استفاده کنید، مجبور خواهید بود که با زبان VBScript و HTML برنامه بنویسید.

هم‌چنین باید فایل‌های OCX را به فایل cab الصاق کنید که روی سرور قرار گیرند. ویزارد Package and Deployment ابزار خیلی مفیدی برای این عملیات است.

۶ــ۴ــ کاربرد File System Objects

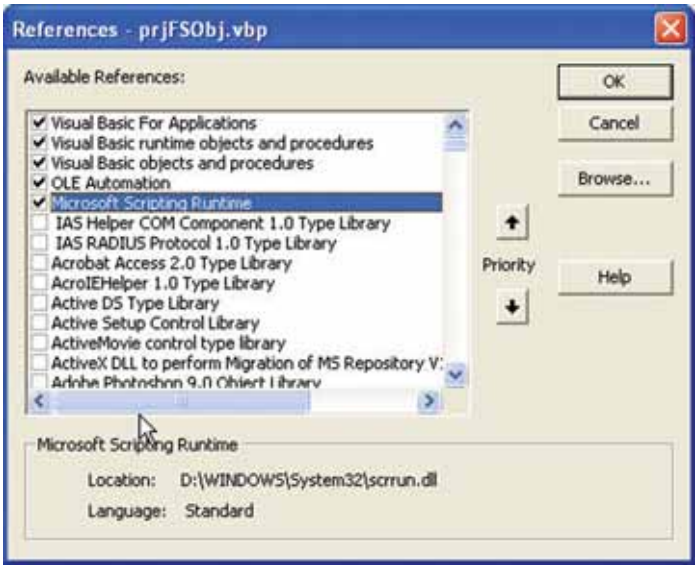
نکته: شیءهای سیستم فایل در مقابل File System Objects

شیءهای سیستم فایل، تمام شیءهایی هستند که امکان کار کردن با سیستم فایل کامپیوتر را فراهم می‌کنند : Drive ، Folder ، File ، TextStream و FileSystemObject .

شیء FileSystemObject یک شیء مرکزی است که امکان دسترسی به شیءهای مختلف سیستم فایل را براساس مقادیری که متدهای این شیء برمی‌گردانند، فراهم می‌کند.

قبل از به‌کار بردن شیءهای سیستم فایل که شیءها، متدها و داده‌های مورد نیاز با فایل‌ها، پوشه‌ها و درایوهای داخل سیستم فایل کامپیوتر را کپسوله‌سازی کرده‌اند، باید DLL مربوطه را به IDE ویژوال بیسیک اضافه کنید. بعد از باز کردن کادر محاوره‌ای References (گزینه‌ی References

از منوی Project)، گزینه‌ی Microsoft Scripting Runtime را انتخاب کنید. این عمل شبیه اضافه کردن شیء COM به IDE ویژوال بیسیک است (شکل ۲۶ــ۴).



شکل ۲۶ــ۴ــ می‌توان از شیءهای سیستم فایل برای دسترسی به متدها و شیءهای مورد نیاز برای کار کردن با فایل‌های روی دیسک، استفاده کرد.

جدول ۳ــ۴، شیءهای مختلف سیستم فایل را نشان می‌دهد.

جدول ۳ــ۴ــ شیءهای سیستم فایل

نام	شرح
Drive	امکان دسترسی به درایوهای مختلف در روی سیستم را فراهم می‌کند. این درایوها می‌توانند درایوهای CD-ROM ، RAM Disks یا درایوهای شبکه باشند.
Folder	امکان کار کردن با پوشه‌های روی سیستم را فراهم می‌کند. می‌توان نام و محل آن‌ها را به دست آورد. هم‌چنین می‌توان پوشه‌ها را با متدهای شیء Folder ، ایجاد و حذف کرد.
File	امکان باز کردن، ایجاد یا جابه‌جایی فایل‌ها را می‌دهد.
FileSystem Object	شیء سیستم فایل مرکزی است. از FileSystemObject برای دسترسی به سایر شیءهای سیستم فایل استفاده کنید.
TextStream	امکان خواندن، نوشتن و اضافه کردن به فایل‌های متنی را فراهم می‌کند.

یک FileSystemObject را با استفاده از کلید واژه‌ی New اعلان کنید. بنابراین برای اعلان یک fso ، از کد زیر استفاده کنید :

```
Dim fso As New File System object
```

هنگامی که یک FileSystemObject دارید می‌توانید از آن شیء برای دسترسی به سایر شیءهای سیستم فایل مثل شیء TextStream استفاده کنید. مثال زیر، چگونگی استفاده از این شیء را نشان می‌دهد.

مثال ۱۰ــ۴ــ شیء TextStream محتوای فایل‌های متنی را به صورت یک رشته‌ی خیلی طولانی می‌خواند. بنابراین می‌توان از خواندن خط به خط محتوای فایل متنی، پرهیز کرد. کد زیر، چگونگی این عمل را نشان می‌دهد. درخط ۲۱ از متد Read شیء ts از نوع TextStream برای خواندن محتوای فایل متنی در داخل حلقه استفاده شده است (خطوط ۲۰ تا ۲۲).

```
01 Dim fso As New File System object
02 Dim ts As TextStream
03 Dim strData As String
04
05 `Set a common dialog filter to show only
06 `text files
07 cdlgMain. Filter = Text(*.txt)|*.txt''
08
09 `Open the common dialog in show Open mode
10 cdlgMain.ShowOpen
11 gf_strOpenFile =cdlgMain.FileName
12
13 `Get a Text Stream Object using the
14 `Open TextFile method of the File System Object
15 `Set ts = fso.Open TextFile(gf_strOpenFile)
16
17 `Traverse to the end of the TextStream
```