مسأله، تعریف یک بسته‌ی متغیری و نامگذاری آن است. این بسته را نوع داده‌ی تعریف شده به‌وسیله‌ی کاربر می‌نامند. متغیرهای UDT را نیز با استفاده از کلید واژه‌های Dim، Public یا Private می‌توان اعلان کرد.

یک UDT را می‌توان با استفاده از کلید واژه‌ی Type در بخش General مدول ایجاد کرد :

```
Type TypeName
    Elements as DataType
    .....
End type
```

در این دستور :

- Type : کلید واژه‌ی ویژوال بیسیک است که شروع یک بلاک نوع داده را مشخص می‌کند.
- *TypeName* : نام نوع داده است.
- Elements as DataType : هرعضوی از نوع داده است.
- End type : پایان دستور Type را مشخص می‌کند.

شکل کلی رجوع به عناصر نوع داده‌ی تعریف شده به وسیله‌ی کاربر به صورت زیر است :

```
VarName.ElementName
```

که در این شکل کلی :

- VarName : نام متغیر است (نمونه‌ای از نوع داده).
- ElementName : نام عنصر خاصی از نوع داده است.

کد زیر، چگونگی ایجاد نوع داده‌ی Music که از اجزای قطعه‌ی موسیقی و آهنگساز تشکیل شده است را نشان می‌دهد. همچنین در این کد، متغیری به نام MyMusic از نوع داده تعریف شده است.

```
01 `Make a user-defined type that has
02 `elements for the composer and the piece
03 Type Music
04 Composer As String
05 Piece As  String
```

```
06 End Type

01 Dim MyMusic As Music
02 Dim Msg$
03 `Assign values to each element in the user-
04 `defined type. Values come from TextBoxes on a form.
05 MyMusic. Composer = txtComposer. Text
06 MyMusic.Piece = txtPiece. Text
07
08 `Create a string that displays all of the
09 `values in the type
10 Msg$ ="Composer: "&MyMusic.Composer & vbCrLf
11 Msg$ = Msg$ & "Piece: "& MyMusic. Piece
12
13 `Display the string
14 MsgBox Msg$
```

متغیرهای نوع داده را نیز می‌توان به صورت محلی یا عمومی تعریف کرد.

### ۱ــ۶ ــ۱ــ باز کردن فایل‌های تصادفی

قبلاً دیدید که در حالت پیش‌فرض، دستور Open فایل‌ها را در حالت تصادفی باز می‌کند :

```
Open “Random.txt” As#1
```

که معادل دستور زیر است :

```
Open “random.txt” For random As#1
```

(البته می‌توان یک فایل را در حالت تصادفی باز کرد ولی با آن به صورت ترتیبی کار کرد.)

برای درک بهتر تفاوت فایل‌های ترتیبی و تصادفی به یک مثال توجه کنید. فرض کنید فایلی دارید که در ۱۰ خط آن مقدار کل کالاهای یک انبار را نوشته‌اید. حال اگر این فایل را در حالت ترتیبی باز کرده باشید و بخواهید خط ششم (رکورد ششم) را بخوانید باید پنج رکورد قبل را هم بخوانید تا بتوانید به رکورد ششم دسترسی پیدا کنید. ولی در حالت تصادفی، می‌توانید به‌طور مستقیم به سراغ

رکورد ششم رفته و آن را بخوانید (در نوشتن فایل هم همین مطلب صادق است). وقتی فایلی دارای ۱۰ کورد باشد، تفاوت چندان محسوس نخواهد بود ولی می‌توانید تصور کنید که برای فایلی با ۱۰۰۰۰ رکورد چه تفاوتی بین این دو روش وجود خواهد داشت.

### ۲ــ۶ ــ۱ــ دستورهای Get و Put

برای خواندن و نوشتن فایل‌های تصادفی از دو دستور #Get و #Put استفاده می‌کنیم. این دو دستور معادل دستورات #Input و #Print در فایل ترتیبی هستند. ولی تفاوت کوچکی بین این دستورها وجود دارد. در دستورهای #Input و #Print راهی وجود ندارد تا نقطه‌ای از فایل برای خواندن یا نوشتن مشخص شود، در حالی که دستورهای #Get و #Put دارای چنین امکانی هستند :

```
Put [#]intFileNum, [intRecNum,] Variable
Get [#]intFileNum, [intRecNum,] Variable
```

- intFileNum شماره‌ی فایل موردنظر است.
- intRecNum شماره‌ی رکوردی است که می‌خواهید با آن کار کنید.

> **نکته:** در صورتی که شماره‌ی رکورد در این دستورها ذکر نشود، عملیات روی رکورد بعد از رکورد جاری انجام می‌شود.

- Variable متغیری است که مقدار آن در فایل نوشته شده و یا از فایل خوانده شده و در آن ذخیره می‌شود.

همان‌طور که مشاهده می‌کنید در این دستورها از آرگومانی به نام شماره‌ی رکورد استفاده می‌شود. با تعیین شماره‌ی رکورد، می‌توانید فقط آن رکورد را خوانده یا در آن بنویسید. شماره‌ی رکوردها از ۱ شروع می‌شود. این دستورها می‌توانند هر نوع داده‌ای (حتی آرایه یا نوع داده‌ی کاربر) را بخوانند و این قوی‌ترین ویژگی فایل‌های تصادفی است. در مطالب‌بعد طی چند مثال با روش کار این‌ها بیشتر آشنا خواهید شد.

### ۳ــ۶ ــ۱ــ نوع داده‌ی تعریف شده به‌وسیله‌ی کاربر

قبلاً با آرایه‌ها و متغیرهای Visual Basic آشنا شدید. اما باید بدانید که خودتان هم می‌توانید با ترکیب داده‌های ذاتی Visual Basic، انواع داده‌ی جدیدی ایجاد کنید. به این انواع داده، گاهی ساختار (structure) یا رکورد (record) نیز گفته می‌شود.

فرض کنید برنامه‌ای دارید که نیاز به یک دفترچه آدرس دارد. هر آدرس تشکیل می‌شود از نام، نام خانوادگی، آدرس، شماره تلفن و اطلاعاتی از این قبیل. برای کار با این اطلاعات می‌توانید از متغیرهای جداگانه استفاده کنید ولی در این صورت برنامه‌نویسی بسیار خسته کننده و پیچیده خواهد شد. بسیار ساده‌تر خواهد بود تا بتوانیم با ترکیب این متغیرها نوع داده‌ی جدیدی ایجاد کنیم و از آن به بعد با این نوع داده کار کنیم. به این نوع داده، نوع داده‌ی تعریف شده به‌وسیله‌ی کاربر[1] (یا به‌طور اختصار، نوع داده‌ی کاربر) گفته می‌شود. برای تعریف نوع داده‌ی کاربر، از دستور Type استفاده می‌شود. هر نوع داده‌ی کاربر باید دارای یک نام باشد (*TypeName*). این نام باید در کل برنامه منحصر به فرد باشد. تمام انواع داده کاربر باید در سطح مدول تعریف شوند و تعریف آن‌ها در داخل روال‌ها مجاز نیست. اگر یک نوع داده کاربر در مدول فرم تعریف شود، حتماً باید به صورت Private تعریف شده باشد. از طریق کد زیر، می‌توانید مطالب بیشتری درباره‌ی دستور Type یاد بگیرید.

```
1: ‘Module Page of the Project
2: Type UserType
3: strFName As String
4: strLName As String
5: End Type
6: Public Names As UserType
```

در این کد یک نوع داده کاربر به نام UserType تعریف شده است. این نوع داده‌ی جدید دارای دو متغیر از نوع رشته‌ای (strFName و strLName) است. در خط ۶ هم یک متغیر از این نوع جدید تعریف شده است. دقت کنید که UserType یک متغیر نیست بلکه یک نوع داده است و این Names است که متغیری از نوع UserType می‌باشد.

برای دسترسی به اجزای درونی نوع داده کاربر باید از عملگر نقطه (.) استفاده کنیم. به کد زیر توجه کنید :

```
Names.strFName = "Mohammad"
Names.strLNames = "Yamaghani"
lblFName.Caption = "First Name: "&Names.strFName
lblLName.Caption = "Last Name: "&Names.strLName
```

---

۱ـ User Defined Data Type (UDT)

به هر عضو درون نوع داده‌ی کاربر یک فیلد (field) گفته می‌شود. متغیرهای رشته‌ای درون نوع داده‌ی کاربر را می‌توان با گزینه‌ی *StringLenght بعد از As String محدود کرد. بدین ترتیب طول رشته با مقدار ثابت StringLenght مقداردهی خواهد شد. هنگام نوشتن نوع داده‌ی کاربر در یک فایل (به دلیل این‌که اندازه‌ی هر رکورد باید مشخص باشد) فیلدهای رشته‌ای باید اندازه‌ی ثابت داشته باشند. در کد زیر، شکل تغییریافته کد قبل را مشاهده می‌کنید.

```
1: ‘Module Page of the Project
2: Type UserType2
3:      strFName As String * 8
4:      strLName As String * 20
5: End Type
6: Public Names As UserType2
```

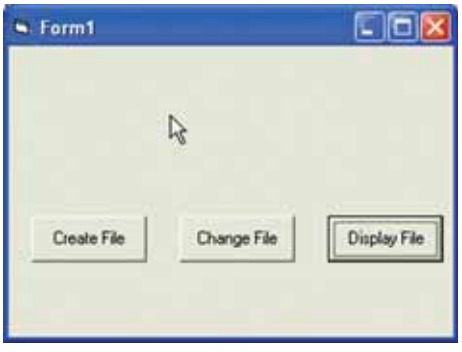
یک رشته‌ی ثابت نمی‌تواند بیش از حداکثر تعیین شده مقدار بگیرد و اگر مقداری که به آن می‌دهیم کمتر از حداکثر طول تعریف شده باشد، Visual Basic بقیه‌ی رشته را با فضای خالی پر خواهد کرد.

**مثال ۵ــ۱ ــ** در کد زیر، اصول کار با فایل‌های تصادفی را مشاهده می‌کنید.

```
1: Private Sub cmdCreate_Click()
2:      ‘This procedure creates the file
3:      Dim intFile As Integer                    ‘Free file number
4:      Dim intCtr As Integer                     ‘Loop countr
5:
6:      intFile = FreeFile
7:      Open “c:\Random.Txt” For Random As#intFile Len = 5
8:
9:      ‘Loop though numbers and with file
10: For intCtr = 1 To 5
11:      Put # intFile, intCtr, intCtr       ‘Record#same as data
12: Next intCtr
```

```
13:
14: Close# intFile
15: End Sub
16:
17: Private Sub cmdChange_Click()
18: ‘This procedure changes 3rd record
19: Dim intFile As Integer          ‘Free file number
20:
21: intFile = FreeFile
22: Open “c:\Rendom.Txt” For Random As #intFile Len = 5
23:
24:     ‘Write a new 3rd record
25: Put #inFile, 3, 9 ، ‘Record3, Value:9
26: Close # intFile
27: End Sub
28:
29: Private Sub cmdDisplay_Click()
30: ‘This procedure displays the file
31: Dim intFile As Integer                    ‘Free file number
32: Dim intVal As Integer                     ‘Read value
33: Dim intCtr As Integer                     ‘Loop counter
34: Dim intMsg As Integer                     ‘For MsgBox()
35: intFile = FreeFile
36: Open “c:\Random.Txt” For Random As #intFile Len = 5
37:
38: intMsg = MsgBox (“File Random. Txt opened...”)
39:
```

```
40: ‘Loop through records and write file
41: For intCtr = 1 To 5
42:       Get #intFile, intCtr, intVal
43: intMsg = MsgBox (“Retrieved a “& intVal & from Random.Txt”)
44: Next intCtr
45: Close # intFile
46:
47: intMsg = MsgBox (“File Random.Txt is now closed”)
48: End Sub
```




شکل ۶ ــ۱

به گزینه‌ی Len در خط ۷ دقت کنید. با این آرگومان طول هر رکورد فایل random.txt معادل ۵ خواهد بود. تعیین طول رکورد بسیار مهم است. اگر طول رکورد معلوم نباشد، دستورات Put و Get نمی‌توانند به درستی کار کنند. (رابطه‌ی یافتن ابتدای یک رکورد خاص در فایل چنین است : شماره‌ی رکورد × طول رکورد.)

برنامه‌ی فوق، دارای سه دکمه‌ی فرمان است. اولی فایل را می‌سازد، دومی آن را تغییر می‌دهد و سومی فایل را نمایش می‌دهد. برای رویداد Click این دکمه‌ها در کد فوق، روال‌های مناسب نوشته شده است. برنامه را ایجاد و اجرا کنید. روی دکمه‌ی Create کلیک کنید تا فایل به‌وجود آید. سپس با کلیک کردن روی دکمه‌ی Display محتویات فایل به‌وجود آمده را مشاهده کنید. حال اگر روی دکمه‌ی Change کلیک کنید، رکورد سوم فایل Random.txt تغییر خواهد کرد. با کلیک کردن دوباره روی دکمه‌ی Display می‌توانید از صحت کار مطمئن شوید. دقت کنید که در خط ۲۵،

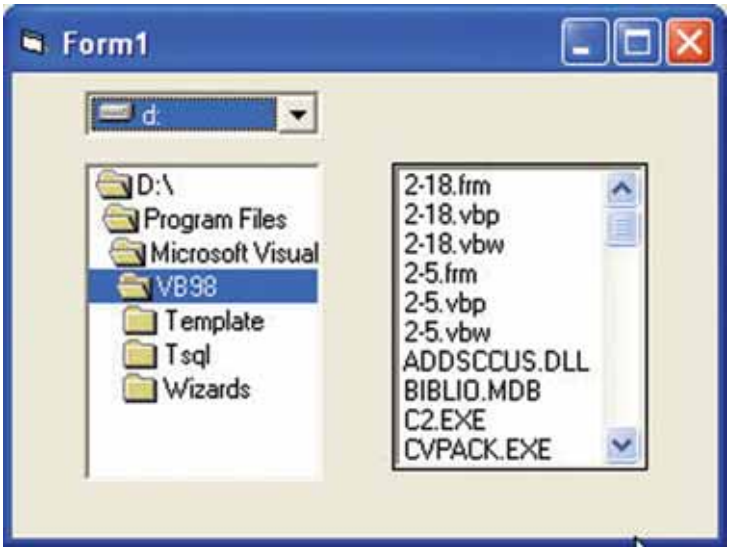
مقدار رکورد سوم فایل به ۹ تغییر داده شده است. مشاهده می‌کنید که برای تغییر این رکورد، نیازی به خواندن تمام فایل نیست.

## ۷ــ۱ــ کنترل‌های FileListBox، DirectoryListBox و DriveListBox

ویژوال بیسیک دارای سه کنترل خاص برای مدیریت پوشه‌ها، درایوها وفایل‌هاست :

- کادر لیست دایرکتوری ــ به کاربر امکان باز کردن دایرکتوری (پوشه)ها را می‌دهد.
- کادر لیست درایو ــ به کاربر امکان انتخاب یک درایو دیسک را می‌دهد.
- کادر لیست فایل ــ به کاربر امکان انتخاب فایل را می‌دهد.

شکل ۷ــ۱ فرمی را با سه کنترل مزبور نشان می‌دهد.

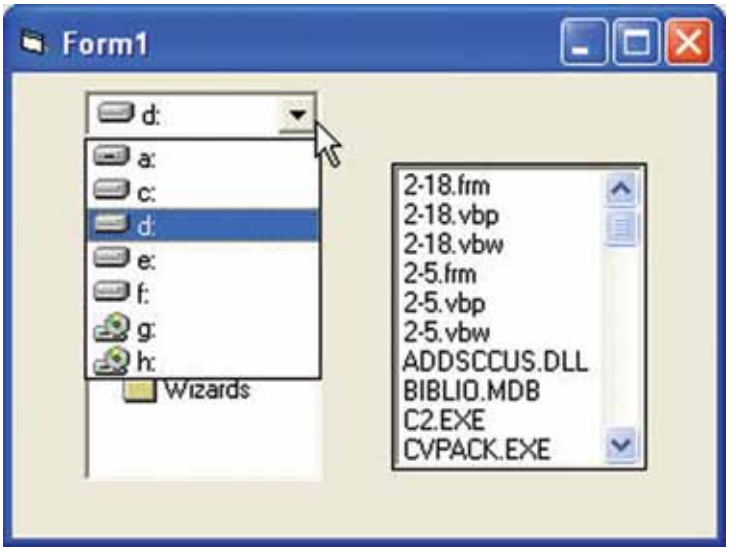



**شکل ۷ــ۱ــ کنترل‌های فایل Visual Basic**

شاید تعجب کنید که با وجود کنترل‌های کادر محاوره‌ای، دیگر چه نیازی به این سه کنترل داریم. مواقعی پیش می‌آید که فقط به یک یا دو جنبه از این کنترل‌ها نیاز داریم. به عنوان مثال، نوشتن فایل‌هایی که دایرکتوری یا درایو آن از قبل مشخص شده‌است (در شبکه‌ها این وضعیت اغلب اتفاق می‌افتد). کنترل‌های فایل در Visual Basic با هم ارتباط ساختاری ندارند، یعنی اگر در کادر لیست درایو، یک درایو را انتخاب کنید، دو کنترل دیگر به طور خودکار متوجه این تغییر نخواهند شد. این وظیفه‌ی برنامه‌نویس است که با نوشتن کد بین آن‌ها ارتباط برقرار کند.

### ۱ــ۷ــ۱ــ کادر لیست درایو

کادر لیست درایو (DriveListBox) به کاربران امکان انتخاب یک درایو را می‌دهد. این کنترل می‌تواند تمام درایوهای موجود در سیستم (دیسک‌های سخت محلی و شبکه، درایوهای فلاپی و CD-ROM) را شناسایی کند. شکل ۸ ــ ۱ یک کادر لیست درایو باز را نشان می‌دهد.




شکل ۸ ــ ۱ــ کادر لیست درایو

درایو پیش‌فرض در کادر لیست درایو، درایوی است که برنامه از آن‌جا اجرا شده است، اما با نوشتن کد می‌توان این حالت را تغییر داد.

مشخصه‌ی متداول این کنترل، Drive است که نام درایو انتخاب شده در آن قرار می‌گیرد. به کمک این مشخصه می‌توان نام درایو پیش‌فرض را نیز تغییر داد.

### ۲ــ۷ــ۱ــ کادر لیست دایرکتوری

با کادر لیست دایرکتوری (DirectoryListBox)، کاربر امکان انتخاب پوشه‌ی موردنظر را خواهد داشت. این کنترل تمام پوشه‌های موجود در سیستم را شناسایی می‌کند. کادر لیست دایرکتوری برای نمایش پوشه‌ها از استانداردهای ویندوز استفاده می‌کند. به یاد داشته باشید که این کنترل نمی‌تواند به طور خودکار درایو انتخاب شده را تشخیص دهد. در این فصل، مشاهده خواهید کرد که چگونه می‌توان بین این کنترل‌ها ارتباط برقرار کرد.

دایرکتوری پیش‌فرض کادر لیست دایرکتوری، پوشه‌ای است که برنامه از آن‌جا اجرا شده است، ولی این وضعیت را می‌توان تغییر داد.

مشخصه‌ی متداول این کنترل، Path است که نام درایو انتخاب شده در آن قرار می‌گیرد. به کمک این مشخصه می‌توان نام پوشه‌ی پیش‌فرض را نیز تغییر داد.

### ۳ــ۷ــ۱ــ کادر لیست فایل

به کمک کادر لیست فایل (FileListBox) کاربر می‌تواند فایل موردنظرش را انتخاب کند. این کنترل قادر است تمام فایل‌های موجود در سیستم را شناسایی کند. این کنترل برای نمایش فایل‌ها از استاندارد گرافیکی ویندوز استفاده می‌کند. به یاد داشته باشید که این کنترل به خودی خود قادر به تشخیص درایو و پوشه‌ی انتخاب شده نیست. کادر لیست فایل به‌طور پیش‌فرض فایل‌های موجود درپوشه‌ای که برنامه از آن‌جا اجرا شده را نمایش خواهد داد. این وضعیت با نوشتن کد مناسب قابل تغییر است.

مشخصه‌های متداول این کنترل در جدول ۳ــ۱ ارایه شده‌اند.

**جدول ۳ــ۱**

| مشخصه | توضیح |
|---|---|
| FileName | نام فایل انتخاب شده |
| Path | مسیر فایل انتخاب شده |
| Pattern | تعیین نوع فایل‌های قابل مشاهده در این کنترل |
| Multi Select | امکان انتخاب چندین فایل به صورت همزمان |

## ۸ ــ۱ــ دستورهای فایل

ویژوال بیسیک دارای چندین دستور برای کار با فایل‌ها، پوشه‌ها و درایوها است. این دستورات را در جدول ۳ــ۲ ملاحظه می‌کنید.

علاوه بر دستورهای جدول ۳ــ۲، Visual Basic از توابع Dir() (برای تعیین وجود یاعدم وجود فایل‌ها) و CurDir() (برای تعیین نام دایرکتوری جاری) نیز پشتیبانی می‌کند. فرض کنید می‌خواهید در شروع برنامه کنترل‌های لیست درایو و دایرکتوری به مسیر C:\MyFiles اشاره کنند. برای این کار باید از کد زیر در روال Form_Load() استفاده کنید :

```
ChDrive"C:"
```

ChDir“\MyFiles”

تابع Dir() به توضیح بیشتری نیاز دارد. فرض کنید می‌خواهید بدانید آیا فایلی به نام

**جدول ۴ــ۱ــ دستورات فایل Visual Basic**

| دستور | مفهوم |
|---|---|
| ChDrive *strDrive* | به درایو strDrive تغییر درایو می‌دهد. |
| ChDir *strDirectory* | به دایرکتوری Strdirectory تغییر دایرکتوری می‌دهد. اگر درایو مشخص نشده باشد، Visual Basic از درایو جاری استفاده خواهد کرد. |
| Kill *strFileSpec* | فایل (یا فایل‌های) تعیین شده را پاک می‌کند. |
| MkDir *strDirectory* | دایرکتوری strDirectory را می‌سازد. |
| RmDir *strDirectory* | دایرکتوری strDirctory را حذف می‌کند. اگر در دایرکتوری هنوز فایلی وجود داشته باشد، این دستور تولید خطا خواهد کرد. |

SALES98.DAT در درایو C وجود دارد یا خیر؟ برای این کار می‌توانید از کد زیر استفاده کنید :

```
If (Dir("C:\SALES98.DAT")) = "SALES98.DAT" Then
        IntMsg = MsgBox ("The file exist")
Else
        IntMsg = MsgBox ("The file does not exist")
End If
```

اگر فایل خواسته شده موجود باشد، این تابع نام آن را برمی‌گرداند و در صورتی که فایل موجود نباشد، تابع Dir() چیزی برنمی‌گرداند. آرگومان تابع Dir() (مانند اکثر توابع فایل) می‌تواند از کاراکترهای عمومی (Wildcard) استفاده کند :

```
Dir("C:\Sales*.DAT")
```

این دستور اولین فایلی را که با آرگومان داده شده مطابقت داشته باشد، برمی‌گرداند. بعد از اجرای اولین دستور Dir می‌توانید از آن به بعد تابع Dir() بدون آرگومان (یا حتی بدون پرانتز) استفاده کنید. در این حالت Visual Basic فایل‌های بعدی که با معیار بالا مطابقت داشته باشند را تا زمانی که رشته‌ی null ("") برگشت داده شود، برخواهد گرداند.

اگر بخواهید درایو پیش‌فرض کادر لیست درایو را تغییر دهید باید از خاصیت Drive استفاده کنید :

```
drvDisk.Drive = "d:\"
```

با این کار، هنگامی که این کنترل فعال شود، درایو D: در بالای آن مشاهده خواهد شد. هنگامی که در این کنترل، کاربر درایو دیگری را انتخاب کند، رویداد Change() رخ خواهد داد که می‌توانید در روال drvDisk_Change() درایو پیش‌فرض را تنظیم کنید :

```
ChDrive drvDisk.Drive
```

در صورتی که می‌خواهید بین کنترل‌های کادر لیست دایرکتوری و کادر لیست درایو ارتباط برقرار کنید، باید به صورت زیر عمل کنید :

```
dirDirect.Drive = drvDisk.Drive
```

با این کار، درایو تعیین شده در کادر لیست درایو تبدیل به درایو پیش‌فرض کادر لیست دایرکتوری خواهد شد. دستور فوق را می‌توانید در روال رویداد drvDisk_Change() قرار دهید.

برای برقراری ارتباط بین کادر لیست دایرکتوری و کادر لیست فایل می‌توانید دستور زیر را در روال رویداد Change کادر لیست دایرکتوری قرار دهید :

```
chDir dirDirect.Path
```

کادر لیست دایرکتوری دارای مشخصه‌ی جالبی به نام ListIndex است : مقدار . این مشخصه به اولین زیر دایرکتوری تحت دایرکتوری انتخاب شده اشاره می‌کند. ۱- به خود دایرکتوری اشاره می‌کند. ۲- به یک دایرکتوری بالاتر از آن و الی آخر. اعداد مثبت هم به ترتیب به دایرکتوری‌های پایین‌تر اشاره خواهند کرد.

اگر می‌خواهید کادر لیست فایل فقط انواع خاصی از فایل‌ها را نمایش دهد، می‌توانید از خاصیت Pattern این کنترل استفاده کنید :

```
filFiles.Pattern = “.vbp; *.frm”
```

هنگامی که کاربر فایلی را انتخاب می‌کند، رویداد Change رخ داده و نام فایل انتخاب شده در خاصیت FileName قرار داده می‌شود. این کنترل هم (مانند کادر لیست دایرکتوری) دارای مشخصه‌ی ListIndex است و فایل انتخاب شده مقدار ۱- دارد. روش دیگر ایجاد ارتباط بین کادر لیست دایرکتوری و کادر لیست فایل استفاده از خاصیت Path است :

```
filFiles.Path = dirDirect.Path
```

**تمرین:** پروژه‌ای ایجاد کنید که بتوان با انتخاب یک فایل در کادر لیست فایل، نام فایل و مسیر کامل آن را در یک کنترل برچسب مشاهده کرد.

**تذکر:** در کادر لیست فایل، فقط فایل‌های از نوع TXT و DOC قابل رؤیت باشند.

## ۹ــ۱ــ ذخیره و بازیابی تصویرها

به همان ترتیبی که متن را در یک فایل ذخیره یا از آن بازیابی کردید، می‌توانید تصویرهای طرح بیتی یا فایل آیکون را به کمک تابع LoadPicture() (که قبلاً با آن آشنا شده‌اید) از روی دیسک خوانده و در مشخصه‌ی Picture کنترل کادر تصویر یا تصویر قرار دهید.

همان‌طور که می‌دانید شکل کلی تابع LoadPicture() به صورت زیر است :

```
ImageCtrl.Picture = LoadPicture (FilePath)
```

در این شکل کلی تابع :

- ImageCtrl : کنترل کادر تصویر، تصویر یا فرم است.
- Picture : مشخصه‌ی Picture شیء است.
- LoadPicture : نام تابع است.
- FilePath : محل دقیق فایل روی دیسک است.

برای ذخیره‌ی تصویری که در کنترل‌های کادر تصویر، تصویر یا فرم است، از دستور SavePicture استفاده کنید. شکل کلی این دستور به صورت زیر است :

```
SavePicture Picture, strFilePath
```

در این دستور :

- SavePicture نام دستور است.
- Picture تصویری است که در مشخصه‌ی Picture کنترل‌های کادر تصویر، تصویر یا فرم قرار داده شده است.
- StrFilePath مسیر و نام فایلی است که می‌خواهید تصویر را در آن ذخیره کنید.

**مثال ۶ ــ۱ــ** در این مثال از دستور SavePicture برای ذخیره‌ی تصویر موجود در یک کنترل تصویر (Image) و در محل خاصی روی دیسک، استفاده شده است. این برنامه از یک کادر محاوره‌ای برای تعیین نام فایل و محل ذخیره‌ی آن استفاده می‌کند.

```
01 Private Sub cmdImgSave_Click()
```

```
02 Dim strFilter As String `common dialog filter
03 Dim strFileName As String `Filename variable
04
05 `Set the CommonDialog filter
06 strFilter = "Bitmaps (*.bmp)|*.bmp"
07
08`Assign the filter
09 cdMain.Filter = strFilter
10
11 `Show the dialog
12 cdMain.ShowSave
13
14 `Make sure a value was entered in the
15 `common dialog.
16 If cdMain.filename<> ""Then
17 strFileName = cdMain.filename
18
19 `Save the Picture in the image control
20 SavePicture imgMain.Picture, strFileName
21
22 `Tell the user the file's been saved
23 MsgBox strFileName & "saved."
24 End If
25 End Sub
```

## ۱۰ـــ۱ـــ ذخیره و بازیابی داده‌ها با توابع رجیستری ویژوال بیسیک

می‌توان از رجیستری ویندوز برای ذخیره‌ی اطلاعات کم برنامه استفاده کرد. از رجیستری

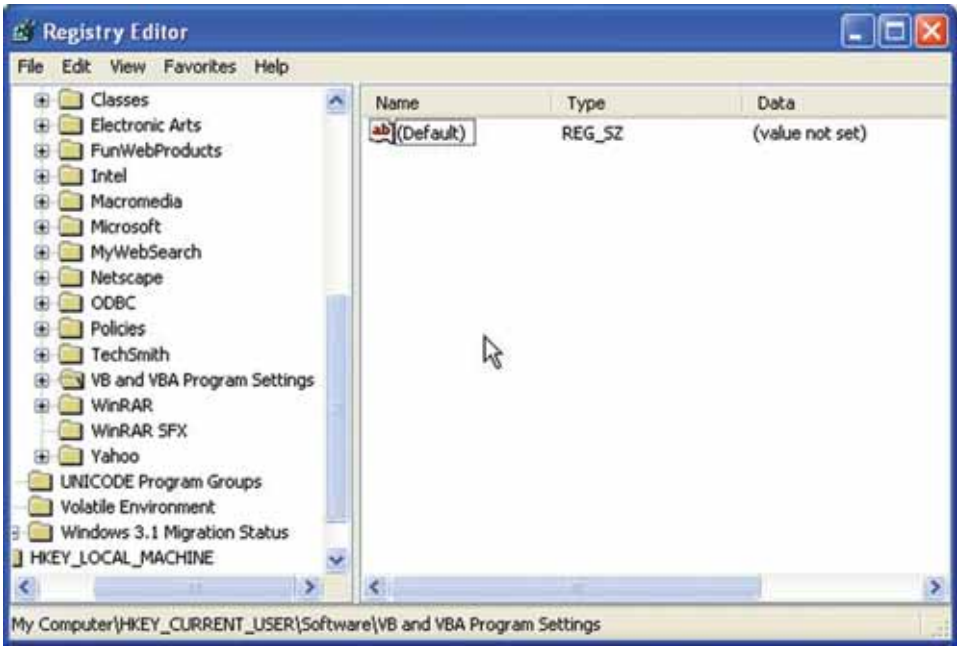
می‌توان برای ذخیره‌ی اطلاعاتی درباره‌ی محل و اندازه‌ی فرم‌های برنامه یا اولویت‌های کاربر استفاده کرد. بعضی از برنامه‌نویسان، رجیستری را برای ذخیره‌ی لیستی از آخرین فایل‌های مورد استفاده به کار می‌برند. مطلب مهم این است که استفاده از رجیستری برای ذخیره‌ی اطلاعات نسبتاً کوچک، روش سریع و ساده‌ای است.

ویژوال بیسیک دارای چهار روال برای دسترسی به رجیستری ویندوز است : DeleteSetting، GetSetting، GetAllSettings و SaveSetting. ویژوال بیسیک می‌تواند عملیات خواندن و نوشتن داده‌ها را در یک کلید خاصی از رجیستری انجام دهد.

(My Computer\HKEY_CURRENT|USER|Software\VB and VBA program Settings ).

این عمل در ویژوال بیسیک به صورت خودکار انجام می‌شود. ویژوال بیسیک نمی‌تواند بدون راهنمایی توابع win32 API روی کلیدهای دیگر رجیستری، عمل خواندن یا نوشتن انجام دهد.

شکل ۹ــ۱، محل کلیدهای کاربردی ویژوال بیسیک در رجیستری ویندوز را نشان می‌دهد.




**شکل ۹ــ۱ــ هنگامی که از توابع رجیستری ویژوال بیسیک استفاده می‌کنید، داده‌ها در کلید تعیین شده برای برنامه‌های کاربردی VB و VBA ذخیره می‌شوند.**

از تابع GetSetting() برای بازیابی مقداری در بخش خاصی از کلید ویژوال بیسیک رجیستری ویندوز استفاده کنید.

MyString = GetSetting (VBKeyName, Section, key [,Default])

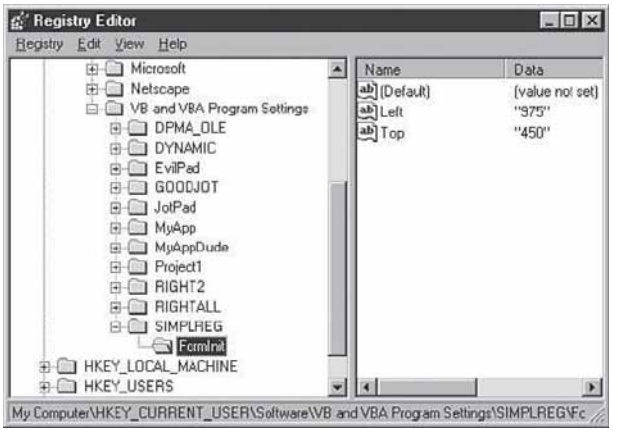
در این شکل کلی تابع :

- MyString رشته‌ی برگردانده شده به وسیله‌ی تابع است.
- GetSetting نام تابع است.
- VBKeyName رشته‌ای است که نام کلید داخل ناحیه‌ی VB\VBA رجیستری را برمی‌گرداند.
- Section رشته‌ای است که بخش یا زیرکلیدی برای تنظیمات خاص برنامه‌ی کاربردی را ارایه می‌دهد.
- Key رشته‌ای است که نام ورودی خاصی داخل بخش را ارایه می‌کند. یک بخش می‌تواند چندین کلید داشته باشد.
- آرگومان اختیاری Default رشته‌ای است که در صورت ناموفق بودن تابع یا بروز خطا برگردانده می‌شود.

بنابراین، خط زیر

Return$ = GetSetting (App.Tittle, “formInit”, “Left”, DefaultLeft$)

ورودی رجیستری که در شکل ۱۰ــ۱ نشان داده شده است را جستجو می‌کند. در صورت ناموفق بودن، مقدار تعیین شده برای رشته‌ی DefaultLeft$ را برمی‌گرداند.




**شکل ۱۰ــ۱ــ توابع رجیستری VB می‌توانند داده‌ها را تحت کلید برنامه ذخیره کنند.**

از دستور SaveSetting برای ذخیره‌ی مقادیر در رجیستری استفاده کنید :

```
SaveSetting VBKeyName, Section, Key, Setting
```

به عنوان مثال :

```
SaveSetting App.Title, "FormInit", "Left", "975"
```

تنظیمات رجیستری نشان داده شده در شکل **۴ــ۷** را به‌وجود می‌آورد.

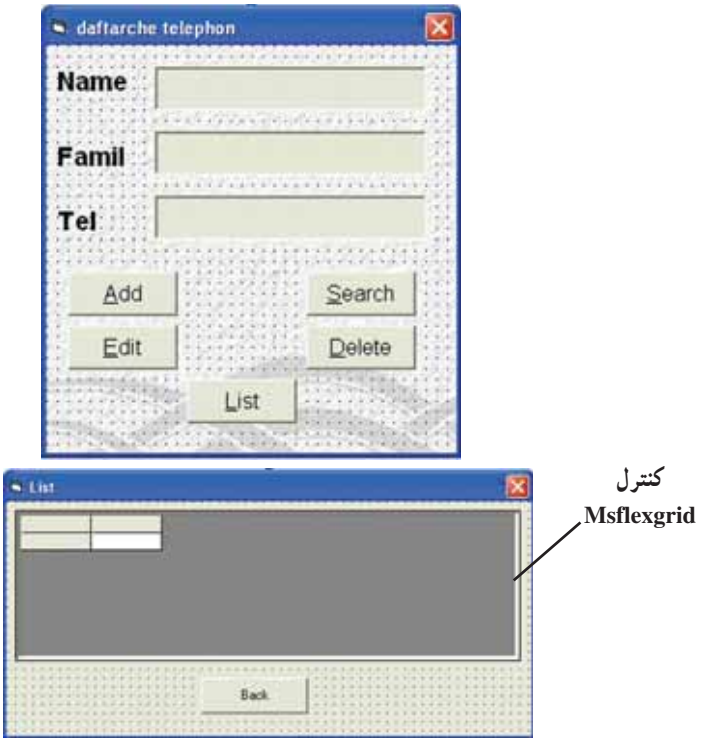
برای به دست آوردن آرایه‌ای از تنظیمات رجیستری، از تابع GetAllSettings() استفاده کنید :

```
MyVariant = GetAllSettings(VBKeyName, Section)
```

در صورتی که می‌خواهید ورودی بخشی از کلید را حذف کنید، دستور DeleteSetting را به کار ببرید :

```
DeleteSetting VBKeyName, Section[,key]
```

**مثال ۷ــ۱ــ دفترچه تلفن:** این مثال، پروژه‌ی کوچکی است که از فایل تصادفی استفاده می‌کند. فرم‌های مربوط به این پروژه را به صورت زیر ایجاد کنید.




شکل ۱۱ــ۱

کد ماژول به صورت زیر است :

```
Public Type TypInfo
        namel As String * 20
        famil1 As String* 30
        tell As String * 15
End Type
Public StrPerson As TypInfo, StrFs As String
Public StrPath As String
```

کد فرم اول به صورت زیر است :

```
Private Sub cmdadd_Click()
Dim LngTotalRec As Long
If Dir (StrFs)<> "" Then
        LngTotalRec = FileLen (StrFs) / Len(StrPerson)
 Else
        LngTotalRec = 0
End If
        Open StrFs For Random Access Write As #1 Len = Len(StrPerson)
        Seek #1, LngTotalRec + 1
        StrPerson.namel =Trim(UCase(InputBox("Enter name:","Add Data")))
        StrPerson . famil1 = Trim (UCase ( InputBox ("Enter famil :", "Add
        Data")))
        StrPerson.tell = Trim(UCase(InputBox("Enter Tel:", "Add Data"))
        Put # 1, , StrPerson
        Close # 1
End Sub
Private Sub cmddelete_Click()
Dim StrPerson As TypInfo
```

```
Dim StrOldName As String, StrOldFamil As String, StrFt As String
Dim IntYn, BlnResult As Boolean
Dim LngTotalRes As Long, LngI As Long
        LngTotalRec = FileLen(StrFs)/Len(StrPerson)
        If LngTotalRec = 0 Then
            MsgBox "No Existing Data"
            Exit Sub
        End If
StrOldName = Trim (UCase ( InputBox ("Enter Name for Delete:", "Delete
 Data")))
StrOld Famil = Trim(UCase(InputBox("Enter Famil for Delete:", "Delete
 Data")))
    IntYn = MsgBox("Are You Sure Erasing Data?", vbYesNo, "Erase
Data")
    If IntYn = vb Yes Then
            StrFt = StrPath + "tel.tmp"
           Open StrFs For Random Access Read As #1 Len = Len(StrPerson)
           If Dir (StrFt)<> ""Then Kill StrFt
           Open StrFt For Random Access Write As #2 Len = Len(StrPerson)
           BlnResult = False
              For LngI = 1 To LngTotalRec
              Get # 1,, StrPerson
            If Not (Trim(StrPerson.famill) = StrOldFamil And
            Trim(StrPerson.namel) = StrOldName) Then
                                    Put #2, ,StrPerson
                                  Else
                                      BlnResult = True
```

```
            End If
        Next
        Close #1, #2
        If BlnResult = True Then
            Kill StrFs
            Name StrFt As StrFs
            MsgBox "Eraseing Information Successfuly"
        Else
            Kill StrFt
            MsgBox "Can Not Erase Information"
              End If
        End If
    End Sub
Private Sub cmdedit_Click()
        Dim StrOldName As String, StrOldFamil As String
        Dim StrNewName, StrNewFamil, StrNewTel As String
        Dim LngTotalRec As Long, LngI As Long, BlnResult As Boolean
            If Dir(StrFs) = ""Then
                MsgBox "Data File Not Found!"
                Exit Sub
            End If
            LngTotalRec = FileLen(StrFs)/ Len(StrPerson)
            If LngTotalRec = 0 Then
                MsgBox "No Existing Data"
                Exit Sub
            End If
            StrOldName = UCase (InputBox("Enter Old Name:", "Edit"))
```

```
        StrOldFamil = UCase (InputBox("Enter Old Famil:", "Edit"))
        StrNewName = UCase(InputBox("Enter New Name:", "Edit"))
        StrNew Famil = UCase (InputBox("Enter New Famil:"))
        StrNewTel = UCase (InputBox("Enter New Tel:"))
        Open StrFs For Random Access  Read Write As #1 Len = Len(StrPerson)
        BlnResult = False
        For LngI = 1 To LngTotalRec
            Get #1,, StrPerson
            If Trim(StrPerson.famill)=Trim(StrOldFamil)And Tim(StrPerson.name 1)
= Trim(StrOldName) Then
            StrPerson.famill = StrNewFamil
            StrPerson.namel = StrNewname
            StrPerson.tel1 = StrNewTel
            Put#1, LngI, StrPerson
            lblfamil.Caption = StrPerson.famill
            lblname.Caption = StrPerson.namel
            lbltel.Caption = Strperson.tell
            BlnResult = True
        End If
    Next
    Close #1
        If BlnResult = True Then
        MsgBox "Editing Successfuly"
    Else
        MsgBox "Not Found Data For Editing"
    End If
    End Sub
```

```
Private Sub CmdList_Click()
        FrmList.Show 1
End Sub

Private Sub cmdsearch_Click()
Dim StrsearchFamil As String
Dim StrPerson As TypInfo
Dim LngTotalRec As Long, LngI As Long
If Dir(StrFs) = ""Then
        MsgBox "Data File Not Found!"
        Exit Sub
End If
LngTotalRec = FileLen(StrFs) / Len(StrPerson)
If LngTotalRec = 0 Then
        MsgBox "No Existing Data"
        Exit Sub
End If
StrSearchFamil = Trim(UCase(InputBox("Enter Famil For search:")))
Open StrFs For Random Access Read As #1 Len = Len(StrPerson)
For LngI = 1 To LngTotalRec
        Get # 1,, StrPerson
        If Trim(StrPerson.famill) = StrSearchFamil Then
            lblfamil.Caption = StrPerson. famill
            lblname. Caption = StrPerson.namel
            lbltel.Caption = StrPerson.tell
        End If
```

```
    Next
    Close #1
End Sub

Private Sub Form_Load()
StrPath = App.Path
If Right (StrPath, 1) <>"\" Then StrPath = StrPath +"\"
StrFs = StrPath + "tel.dat"
End Sub
```

کد فرم دوم به صورت زیر است :

```
Private Sub CmdBack_Click()
        Me.Hide
End Sub

Private Sub Form_Activate()
Dim LngI As Long
With msfl
        .Rows = 1
        .Cols = 4
        .Clear
        .ColWidth(0) = 500
        .ColWidth (1) = 2000
        .ColWidth (2) = 2000
        . ColWidth (3) = 2000
        . Width =.ColWidth(0)+ .ColWidth(1) +. ColWidth(2)+
.ColWidth(3)+50
```

```
        . Row= 0
        . Col = 1
        . Text ="Name"
        . Col = 2
        . Text = "Famil"
        .Col = 3
        . Text = "Tel"
If Dir(StrFs) = "" Then
        MsgBox "Data File Not Found!"
        Exit Sub
End If
Open StrFs For Random Access Read As #1 Len = Len(StrPerson)
For LngI = 1 To FileLen (StrFs) \ Len(StrPerson)
        Get #1,, StrPerson
        .AddItem LngI
        . Row = LngI
        . Col = 1
        . Text = Trim(StrPerson.famill)
        .Col = 2
        .Text = Trim (StrPerson.namel)
        .Col =3
        .Text = Trim(StrPerson.tell)
Next
Close#1
End With
End Sub
```

## خودآزمایی و تحقیق

۱ــ برنامه‌ای بنویسید که مجموعه‌ای از اعداد صحیح را از فایل data.dat بخواند و به صورت صعودی آن‌ها را مرتب کند.

۲ــ خطاهای عبارت‌های زیر را پیدا کرده و اصلاح کنید.

الف)
```
Get#6, udtCarInformation    'Store data in record
```

ب)
```
Open #99 For Random Access Append Len = 140
```

ج)
```
Put #33, 15, inventory%    'inventory is an array name
```

د)
```
'Open a file for reading and Writing
Open "c:\customer.rnd" Access Read+
```

۳ــ مجموعه‌ای از دستورها برای فراهم کردن خواسته‌های زیر بنویسید فرض کنید رکورد

```
Type Person
        lastName As String *15
        firstName As String *15
        age As String *3
    End Type
```

از قبل تعریف شده است و فایل با دسترسی تصادفی به درستی باز شده باشد.

الف) برای ۱۰ رکورد داده وارد کرده و آن‌ها را در فایل بنویسید.

ب) اطلاعاتی از رکوردها را به هنگام کنید.

ج) یکی از رکوردها را حذف کنید.

۴ــ فرض کنید شما صاحب یک فروشگاه ابزارآلات هستید و نیاز دارید همواره لیستی از موجودی ابزارهای متفاوت، تعداد، هزینه‌ها و قیمت در اختیار داشته باشید. برنامه‌ای بنویسید که با استفاده از یک فایل تصادفی به نام hardward.dat که دارای صد رکورد خالی است به شما امکان دسترسی به اطلاعات موردنیاز هر کدام از ابزارها را فراهم آورد. این برنامه باید به شما اجازه‌ی لیست گرفتن از تمامی ابزارها، حذف ابزاری که مدت طولانی در اختیار نداشته‌اید و همچنین اجازه به هنگام کردن تمام اطلاعات فایل را بدهد. شماره شناسایی ابزار باید شماره‌ی رکورد باشد. از اطلاعاتی

که در زیر آورده شده است، می‌توانید در فایل خود استفاده کنید.

| شماره رکورد | نام ابزار | تعداد | قیمت به هزار ریال |
|---|---|---|---|
| 3 | Electric sander | 7 | 57.98 |
| 17 | Hammer | 76 | 11.98 |
| 24 | Jigsaw | 21 | 11.00 |
| 39 | Lawn mower | 3 | 79.50 |
| 56 | Power saw | 18 | 99.99 |
| 68 | Sledgehammer | 11 | 21.50 |
| 77 | Screwdriver | 106 | 6.99 |
| 83 | Wrench | 34 | 7.50 |

۵ ــ با یک دستور Close چند فایل را می‌توان بست؟

۶ ــ چه تابعی اولین شماره‌ی فایل آزاد را برمی‌گرداند؟

۷ــ اگر یک فایل ترتیبی را برای خروجی باز کنید و آن فایل موجود باشد، چه اتفاقی خواهد افتاد؟

۸ ــ اگر یک فایل ترتیبی را برای افزودن باز کنید و آن فایل موجود باشد، چه اتفاقی خواهد افتد؟

۹ــ دستور زیر چه نوع فایلی را باز می‌کند؟

```
Open "TestFile.dat" For Append As#1
```

۱۰ــ چرا برای باز کردن فایل‌های تصادفی باید طول هر رکورد معلوم باشد؟

۱۱ــ چرا برای نوشتن نوع داده‌ی کاربر در فایل، طول رشته‌ها باید مشخص باشد؟

۱۲ــ با کدام دستور Visual Basic می‌توانید نوع داده‌ی کاربر را تعریف کنید؟

۱۳ــ تفاوت تابع Dir با آرگومان و تابع Dir بدون آرگومان چیست؟

۱۴ــ هنرجویی برنامه‌ای نوشته و در آن از دستور زیر استفاده کرده است :

```
RmDir"c:\Game"
```

اما اجرای این برنامه با خطا متوقف می‌شود. آیا می‌توانید محتمل‌ترین علت این مشکل را معلوم کنید؟ (فرض کنید دایرکتوری Game در درایو C وجود دارد.)

۱۵ــ روالی بنویسید که یک فایل ترتیبی ایجاد کرده و اطلاعات زیر را در آن بنویسید : نام، سن، رنگ مورد علاقه. پنج رکورد در این فایل قرار دهید (هر رکورد باید دارای یک نام، یک سن و یک رنگ باشد). برای نوشتن در فایل از حلقه‌های For استفاده کنید. راهنمایی : برای هر یک از این مقادیر یک آرایه ایجاد کنید.

۱۶ــ یک کادر محاوره‌ای ایجاد کنید که کادر محاوره‌ای Open ویندوز را شبیه‌سازی کند. فقط از کنترل‌های کادر لیست درایو، دایرکتوری، فایل و دو دکمه‌ی OK و Cancel استفاده کنید. بین سه کنترل درایو، دایرکتوری و فایل ارتباط برقرار کنید.