

آموزش ADO (ActiveX Data Object)

تاریخچه مختصر:

برای آسانتر شدن کار برنامه نویسان و یکسان کردن روشهای ذخیره و بازیابی، استاندارد ODBC پایه گذاری شد. ODBC توابع و روشهای پیچیده ای را برای اتصال به پایگاه داده و استفاده از آن را در اختیار برنامه نویسان قرار می داد. شرکت مایکروسافت مدل برنامه نویسی DAO را ارائه داد. DAO مخفف Database Access Objects می باشد. DAO بر پایه موتور Jet بود ولی با استفاده از اشیاء داده ای که ارائه کرده بود اتصال به بانک و کار با آن را آسانتر کرده بود. پس از آن شرکت مایکروسافت مدل OLEDB را ارائه کرد که از ODBC پیشرفته تر بود و امکان اتصال به منابعی مانند Email و منابع غیر SQL را نیز داشت. کار کردن با توابع OLEDB بسیار سخت و پیچیده بود بنابراین مایکروسافت ADO را که بر مبنای OLEDB است ارائه کرد. ADO مخفف Activex Data Objects می باشد و همه امکانات OLEDB را از طریق Object ها ارائه می کند. ADO در مقابل با DAO اشیاء کمتری دارد و دارای امکانات بیشتری است.

استفاده از ADO:

در این مقاله سعی شده است تا در حد ممکن مطالب بصورت خلاصه شده و مفید ارائه شود. بنابراین هدف توضیح امکانات پیشرفته ADO نیست و یا حداقل فعلاً قصد آموزش مقدماتی ADO را داریم. برای استفاده از ADO دو راه داریم. راه اول استفاده از ADODC یا استفاده از کنترل داده ADO می باشد و راه دوم استفاده از ADODB یا استفاده از توابع و اشیاء ADO است. در این مقاله به دلیل انعطاف بیشتر و امکان استفاده حرفه ای تر، از روش دوم استفاده می کنیم.

اشیا ADO:

در ADO سه شیء اصلی وجود دارد:

- **Connection**: برای اتصال به بانک اطلاعاتی بکار می رود.
- **Recordset**: رکوردست مورد نظر از یک بانک اطلاعاتی را در خود دارد.
- **Command**: برای Stored-Procedure ها بکار می رود.

Connection یک زیرمجموعه به نام Error دارد که خطاهای اتصال به بانک در آن قرار می گیرد. Command هم یک زیرمجموعه به نام Parameter دارد که پارامترهای ارسالی به Stored-Procedure ها را در خود دارد. Recordset یک زیرمجموعه به نام Fields دارد که بیانگر فیلدهای یک

رکوردست است. همه این اشیا و زیرمجموعه های آنها دارای کلکسیون به نام Properties هستند که خواص را در خود دارند.

برای کار با ADO در ویژوال بیسیک ابتدا باید Refrence آن را به پروژه اضافه کنید. برای این کار با استفاده از گزینه Refrences در منوی Project پنجره Refrences را مشاهده می کنید که از لیست درون آن باید گزینه Microsoft Activex Data Objects را انتخاب کنید.

شی Connection:

اولین شیء که در کار با بانک اطلاعاتی به آن نیازمندیم شیء Connection می باشد. این شیء یک اتصال بین برنامه شما و بانک اطلاعاتی برقرار می کند. برای اینکه از شیء Connection در برنامه استفاده کنیم باید نمونه ای از آن را ایجاد کنیم.

```
Dim cnn As New ADODB.Connection
```

با استفاده از متد Open اتصال شیء Connection را با پایگاه داده برقرار می کنیم. در اینجا ذکر یک نکته لازم است و آن انعطاف پذیری بیش از حد اشیا داده ADO است به نحوی که برای استفاده از امکانات آن راههای متنوعی وجود دارد. در اینجا در مورد متد Open دو روش را مثال خواهیم زد اما در قسمتهای دیگر به ذکر یک روش اکتفا خواهیم کرد که لزوماً بهترین روش نیست. حالت کلی متد Open به شکل زیر است:

```
connection. Open ConnectionString,UserID,Password,Options
```

در این متد مهمترین پارامتر آن ConnectionString است. حالت کلی ConnectionString بصورت زیر است:

```
Provider=<Provider Name> ;Data Source=<Source>
```

اگر با ADO نسخه 2.0 و یا 2.1 کار می کنید از Provider Name زیر برای اتصال به بانک اطلاعاتی اکسس استفاده کنید:

```
Provider=Microsoft.Jet.OLEDB.3.51
```

و اگر از نسخه 2.5 و بالاتر استفاده می کنید رشته زیر را بکار ببرید:

```
Provider=Microsoft.Jet.OLEDB.4.0
```

رشته زیر به عنوان ConnectionString برای اتصال به یک بانک اطلاعاتی اکسس به نام test.mdb بکار می رود:

```
Provider=Microsoft.Jet.OLEDB.3.51 ;Data Source=c:\test.mdb
```

لیست Provider Name برای Provider های مختلف به شرح زیر است:

نام Provider	رشته
SQLServer	Provider=SQLOLEDB;Data Source=serverName;" Initial Catalog=databaseName; User ID=username;Password=userPassword;
Oracle	Provider=MSDAORA;Data Source=serverName;User ID=username; Password=userPassword;
Microsoft Jet	Provider=Microsoft.Jet.OLEDB.4.0;Data Source=databaseName;User ID=username;Password=userPassword;
ODBC	Provider=MSDASQL;DSN=dsnName;UID=username;PWD=userPassword;

جزئیات بیشتر درباره هر کدام از پارامترهای آن را می توانید در MSDN بیابید.

اگر بخواهیم برای مثال قبل یک اتصال برقرار کنیم بصورت زیر عمل می کنیم:

```
Dim cnn As New ADODB.Connection
cnn.Open "Provider=Microsoft.jet.oledb.3.51;Data Source=C:\test.mdb"
```

به همین سادگی یک اتصال با بانک اطلاعاتی برقرار کردیم.

شیء Recordset:

برای تشکیل یک رکوردست با استفاده از داده های ذخیره شده در یک بانک استفاده می شود. برای تشکیل یک رکوردست از متد Open شیء Recordset استفاده می شود. حالت کلی آن بدین شکل است:

```
Open (Source, ActiveConnection, CursorType, LockType, Options)
```

در ساده ترین حالت پارامتر Source برابر نام جدولی است که می خواهیم اطلاعات را از آن استخراج کنیم. ActiveConnection هم نام شیء Connection است که در حال حاضر فعال می باشد (بوسیله متد Open اتصال برقرار شده است). در حالتی که Source نام یک جدول باشد پارامتر Options برابر adCmdTable است. دو پارامتر CursorType و LockType را هم در بخش بعدی مفصلاً توضیح خواهیم داد و همچنین حالت های دیگر پارامتر Options و پارامتر Source.

در اینجا اگر فرض کنیم که Test.mdb دارای جدولی است به نام Person با استفاده از کد زیر می توان یک رکوردست از آن را تشکیل داد:

```
Dim cnn As New ADODB.Connection
Dim rst As New ADODB.Recordset

cnn.Open "Provider=Microsoft.Jet.OLEDB.3.51 ;Data Source=c:\test.mdb"
rst.Open "Person", cnn, adOpenKeyset, adLockOptimistic, adCmdTable
.
.
.
cnn.Close
rst.Close
Set cnn = Nothing
Set rst = Nothing
```

در این مثال به دو نکته دقت کنید، اول اینکه پارامترهای CursorType و LockType را به همین ترتیب مقداردهی کنید تا در بخش بعد با آنها بیشتر آشنا شویم. دوم اینکه پس از اتمام عملیات روی داده های رکوردست باید هم Connection و هم Recordset را ببندیم. برای این کار هر دو شیء متدی به نام Close دارند. اما برای اطمینان از، از بین رفتن اشیا آنها را برابر Nothing قرار می دهیم.

اگر فرض کنیم که در جدول Person از بانک Test.mdb یک فیلد با نام Name وجود دارد، کد زیر تمام نامها را استخراج کرده و در یک ListBox با نام lstName قرار می دهد. در این کد از متد MoveNext از شیء Recordset استفاده شده است که باعث می شود یک رکورد در Recordset حرکت کنیم.

```
Private Sub Form_Load()  
    Dim cnn As New ADODB.Connection  
    Dim rst As New ADODB.Recordset  
  
    cnn.Open "Provider=Microsoft.Jet.OLEDB.3.51 ;Data Source=c:\test.mdb"  
    rst.Open "Person", cnn, adOpenKeyset, adLockOptimistic, adCmdTable  
  
    Do While Not rst.EOF  
        lstName.AddItem rst!Name  
        rst.MoveNext  
    Loop  
    cnn.Close  
    rst.Close  
    Set cnn = Nothing  
    Set rst = Nothing  
End Sub
```

Rst.EOF بیانگر انتهای رکوردست است. برای دسترسی به مقدار یک فیلد از نام رکوردست به همراه علامت ! و نام فیلد استفاده می کنیم.

علاوه بر متد MoveNext از متدهای MoveFirst که به اولین رکورد می رود، MovePrevious که به رکورد قبلی می رود، MoveLast به آخرین رکورد می رود، Move به رکورد مورد نظر می رود، می توانید استفاده کنید.

تمرین: با این دانسته ها شما می توانید یک بانک اطلاعاتی برای دفترچه تلفن در اکسس بسازید و داده های آن را هم وارد کنید. سپس در ویژوال بیسیک برنامه ای بنویسید که به این بانک متصل شده و ابتدا همه نامها را در لیست به شما نشان دهد و سپس با کلیک بر روی هر نام مشخصات آن را بصورت مناسب نمایش دهد.

پارامترهای متد Open از Recordset:

پارامتر Source و ActiveConnection را قبلا به اندازه کافی توضیح داده ایم اما پارامتر CursorType مقادیر زیر را می پذیرد:

adOpenForwardOnly: رکوردستی بصورت یک طرفه رو به جلو می سازد. در مواردی که یک رکوردست فقط یک بار پیمایش می شود به دلیل سرعت بالا این نوع رکوردست مناسب است.

adOpenKeySet: یک رکوردست با امکان تغییرات بوسیله کاربران و بصورت دو طرفه ایجاد می کند. شما قادر نیستید به رکوردهایی که توسط کاربران دیگر اضافه شده اند دسترسی پیدا کنید.

adOpenDynamic: شبیه به گزینه قبلی است با این تفاوت که رکوردهای تغییر داده شده و یا اضافه شده بوسیله سایر کاربران توسط شما هم در دسترس هستند.

adOpenStatic: رکوردستی که با این گزینه ساخته می شود بصورت ایستا خواهد بود یعنی تغییرات در این رکوردست اثری ندارد.

پارامتر بعدی در متد Open پارامتر LockType است که می توانید گزینه های زیر را برای آن استفاده کنید:

adLockReadOnly: اجازه دستکاری رکوردست را به شما نمی دهد.

adLockOptimistic: رکوردها هنگام Update رکوردست قفل می شوند.

adLockPessimistic: رکوردها هنگام آغاز ویرایش قفل می شوند.

adLockBatchOptimistic: وقتی متد UpdateBatch روی رکوردست اجرا شود، همه دسته، قفل می شود.

برای پارامتر Options هم گزینه های زیر را استفاده کنید:

adCmdText: مقدار پارامتر Source متن فرمان مورد نظر می باشد مثلاً یک دستور SQL است.

adCmdTable: مقدار پارامتر Source نام جدولی است که همه فیلدهای آن برگردانده خواهد شد.

اضافه کردن و حذف رکورد از رکوردست:

بطور خلاصه برای اضافه کردن رکورد ابتدا متد AddNew از رکوردست را فراخوانی می کنیم و سپس فیلدها را مقدار می دهیم و در آخر متد Update را فراخوانی می کنیم. بهتر است قبل از همه این کارها کرسر را با استفاده از متد MoveLast به آخرین رکورد انتقال دهیم

```
rst.MoveLast  
rst.AddNew  
rst!strName = txtName.Text  
rst!strFamily = txtFamily.Text  
rst!iAge = Val(txtAge.Text)  
rst.Update
```

برای حذف رکورد از رکوردست ابتدا باید کرسر را به رکورد مورد نظر منتقل کنیم برای این کار از حلقه ها استفاده می کنیم. پس از رسیدن به رکورد مورد نظر متد Delete از رکوردست را فراخوانی می کنیم و برای اعمال تغییرات رکوردست روی بانک متد Update را فراخوانی می کنیم:

```

Do While Not rst.EOF
    If rst!strName = "mAm" And rst!strFamily = "BlackGhost" Then
        rst.Delete
        rst.Update
        Exit Do
    End If
Loop

```

استفاده از SQL:

شرمنده! یادگیری SQL به عهده خودتان اما نگران نباشید اینجا فقط از یک SELECT ساده استفاده خواهیم کرد. زمانی که بخواهیم رکوردستی از فیلدهای انتخابی از یک جدول تشکیل دهیم می توانیم از SQL استفاده کنیم. برای این کار دستور SQL را بصورت یک رشته و یا متغیر رشته ای به عنوان پارامتر Source از متد Open ارسال می کنیم و پارامتر آخر آن را هم برابر adCmdText قرار می دهیم. به عنوان مثال:

```

rst.Open "SELECT strName,strFamily FROM tblPerson", cnn, adOpenKeyset, adLockOptimistic, adCmdText

```

و یا بدین صورت

```

Dim strSQL As String
strSQL = "SELECT strName,strFamily FROM tblPerson"
rst.Open strSQL, cnn, adOpenKeyset, adLockOptimistic, adCmdText

```

که روش دوم از خوانایی بیشتری برخوردار است. این مثالها رکوردستی حاوی دو فیلد strName و strFamily تشکیل می دهند

یک مثال دیگر

```

Dim strSQL As String
strSQL = "SELECT * FROM tblPerson WHERE Age=20"
rst.Open strSQL, cnn, adOpenKeyset, adLockOptimistic, adCmdText

```

در این مثال رکوردستی حاوی مشخصات همه افرادی که سن آنها ۲۰ سال است تشکیل داده ایم که همه فیلدهای جدول با استفاده از * در این رکوردست انتخاب شده اند

اگر فرض کنیم که در برنامه ای بخواهیم اطلاعات مثال قبل را بدست آوریم اما سن مورد نظر را از کاربر بپرسیم از طریق زیر عمل می کنیم:

```

Dim strSQL As String
rst.Open strSQL, cnn, adOpenKeyset, adLockOptimistic, adCmdText

```

اگر دقت کرده باشید در این مثالها فرض بر این بوده است که فیلدی که در قسمت شرط در WHERE بکار رفته است از نوع عددی است. اما اگر بخواهیم از فیلدهای رشته ای استفاده کنیم باز هم باید تغییر کوچکی در آن بدهیم و از کدهای زیر استفاده کنیم:

```

strSQL = "SELECT * FROM tblPerson WHERE strName=' " & txtName.Text & " '"

```

در استفاده از دستورات SQL باید به این نکته توجه کنید که در قسمت شرط WHERE باید مقدار بعد از تساوی را بین علامتهای ' ' محصور کنید بنابراین از روش بالا برای این کار استفاده کرده ایم البته برای اینکه این علامت کوچک به راحتی دیده شود دو طرف آن را علامت فاصله قرار داده ایم که این کار را شما نباید انجام دهید

ویرایش رکورد:

ما درباره حذف و اضافه کردن رکوردها در بخش گذشته بحث کردیم و حالا می خواهیم ویرایش رکورد را بررسی کنیم. برای ویرایش رکورد کافی است ابتدا رکوردست مورد نظر را تشکیل دهیم و سپس رکورد را در آن جستجو کنیم و پس از پیدا کردن رکورد فیلدهای آن را به مقدار جدید ست و در آخر رکوردست را Update کنیم. به مثال زیر بدون توجه به منطق آن توجه کنید:

```
Dim cnn As ADODB.Connection
Dim rst As New ADODB.Recordset
.
.
.
Do While Not rst.EOF
    If rst!strName = txtName.Text Then
        rst!Family = txtFamily.Text
        rst!No = txtNo.Text
        rst.Update
        Exit Do
    End If
    rst.MoveNext
Loop
rst.Close
Set rst = Nothing
```

روالهای ذخیره شده (Stored Procedure):

روالهای ذخیره شده در حقیقت دستورات SQL هستند که در قالب Query در بانک اطلاعاتی ذخیره شده اند. یعنی در این روش ما Query های موجود در بانک اطلاعاتی را اجرا می کنیم. این روش نسبت به روش استفاده از رشته SQL در متد Open چندین مزیت دارد:

(a) برای ساختن روالهای ذخیره شده می توانید از ابزار ویزارد بانک اطلاعاتی استفاده کنید. بدین ترتیب می توانید دستورات SQL پیچیده را ظرف مدت زمان کوتاهی ایجاد کنید.

(b) سرعت اجرای روالهای ذخیره شده بیشتر است زیرا این روالها بصورت داخلی اجرا شده و نتیجه آن به عنوان رکوردست برگردانده می شود، در صورتی که روش قبلی در حقیقت فیلتر گذاشتن روی خروجی یک رکوردست می باشد.

(c) خوانایی سهولت استفاده از دستورات SQL پارامتریک که این امر باعث کاهش حجم کد شما خواهد شد.

(d) خوانایی برنامه که از نظر مهندسی نرم افزار امر بسیار مهمی است بسیار بیشتر خواهد بود.

برای اجرای روالهای ذخیره معمولی باید از شی Command و نوع پارامتریک آنها از اشیای Command و Parameter استفاده کرد. بهتر است روش استفاده از شی Command را با استفاده از یک مثال توضیح دهیم. پس به مثال زیر توجه کنید:

```
Dim cnn As New ADODB.Connection
Dim cmd As New ADODB.Command
Dim rst As New ADODB.Recordset

cnn.Open "Provider=Microsoft.jet.oledb.3.51;Data source=" & App.Path & "\Site.mdb"
cmd.ActiveConnection = cnn
cmd.CommandText = "qryInDatabase"
cmd.CommandType = adCmdStoredProc
Set rst = cmd.Execute

Do While Not rst.EOF
    Print rst!AddressName
    rst.MoveNext
Loop
rst.Close
cnn.Close
Set rst = Nothing
Set cmd = Nothing
Set cnn = Nothing
```

در این مثال همه چیز گویاست بجز CommandType و CommandText و Execute. خاصیت CommandType نوع دستور را تعیین می کند که در اینجا قصد استفاده از روال ذخیره شده را داریم. خاصیت CommandText را وقتی مه مقدار خاصیت قبلی را برای استفاده از روال ذخیره شده تعیین کنیم باید برابر نام Query در بانک اطلاعاتی قرار گیرد. متد Execute برای اجرای دستور مورد نظر بکار می رود.

خروجی متد Execute یک شی رکوردست است که با استفاده از دستور Set به یک شی رکوردست نسبت می دهیم. این مثال برای یک Query بدون پارامتر می باشد. برای استفاده از Query های پارامتریک باید از شی Parameter استفاده کنیم

استفاده از شی Parameter:

با یک مثال استفاده از شی Parameter را بررسی می کنیم.

```
Dim cnn As New ADODB.Connection
Dim rst As New ADODB.Recordset
Dim cmd As New ADODB.Command
Dim prm As New ADODB.Parameter
Dim strCon As String

strCon = "Provider=Microsoft.Jet.Oledb.3.51;Data Source=" + App.Path + "\db1.mdb"
'Debug.Print strCon
cnn.Open strCon
cmd.ActiveConnection = cnn
cmd.CommandType = adCmdStoredProc
cmd.CommandText = "qryFamily"
```

```

prm.Name = "Family"
prm.Type = adBSTR
prm.Value = "safdel"

cmd.Parameters.Append prm
Set rst = cmd.Execute

Do While Not rst.EOF
    lstTest.AddItem rst!Name
    rst.MoveNext
Loop

rst.Close
cnx.Close

```

در این مثال فقط قسمتهای مربوط به شی Prm جدید می باشند. خاصیت Name از شی prm نام فیلدی است که بصورت پارامتریک می باشد. Type نوع فیلد را تعیین می کند. خاصیت Value مقدار ارسالی به پارامتر مورد نظر می باشد.

پس از تنظیم خواص لازم از شی prm با استفاده از متد Append از شی Command این پارامتر را به شی Command متصل می کنیم و در نهایت با فراخوانی متد Execute از شی Command رکوردست را تحویل می گیریم.

بدست آوردن اطلاعات از ساختار بانک اطلاعاتی:

با استفاده از متد OpenSchema از شی Connection می توانیم لیستی از جداول موجود در یک بانک اطلاعاتی را بصورت یک رکوردست بدست آوریم.

```

Public Function OpenSchemaX() As String
    Dim Cnxn As ADODB.Connection
    Dim rstSchema As ADODB.Recordset
    Dim strCnxn As String
    Dim strStruct As String

    Set Cnxn = New ADODB.Connection
    strCnxn = "Provider=Microsoft.Jet.Oledb.3.51;Data Source=" + App.Path + "\db1.mdb"
    Cnxn.Open strCnxn

    Set rstSchema = Cnxn.OpenSchema(adSchemaTables)

    Do Until rstSchema.EOF
        strStruct = strStruct + "Table name: " & rstSchema!TABLE_NAME & "Table type: " & rstSchema!TABLE_TYPE + vbCrLf
        List1.AddItem strStruct
        rstSchema.MoveNext
    Loop
    OpenSchemaX = strStruct
    rstSchema.Close
    Cnxn.Close

    Set rstSchema = Nothing
    Set Cnxn = Nothing
End Function

```

```
Private Sub Form_Load()  
    Text1.Text = OpenSchemaX  
End Sub
```

تابع OpenSchemaX لیستی از اجزای بانک اطلاعاتی را بصورت یک رشته برای شما برمی گرداند. و یک نکته دیگر. شما می توانید با استفاده از متد Save از شی رکوردست یک رکوردست را در فایل ذخیره کنید.

مباحث ADO را همینجا تمام می کنیم منتظر نظرات شما هستیم. برای نظر خواهی می توانید از سیستم نظر خواهی وبلاگ که آدرس آن را ذکر کرده ام استفاده کنید.

نویسنده: **مادوگر ویزوال بیسیک**

آدرس وبلاگ: **مادوی ویزوال بیسیک** <http://vbLog.persianblog.com>



پست الکترونیک: mam_programmer@yahoo.co.uk