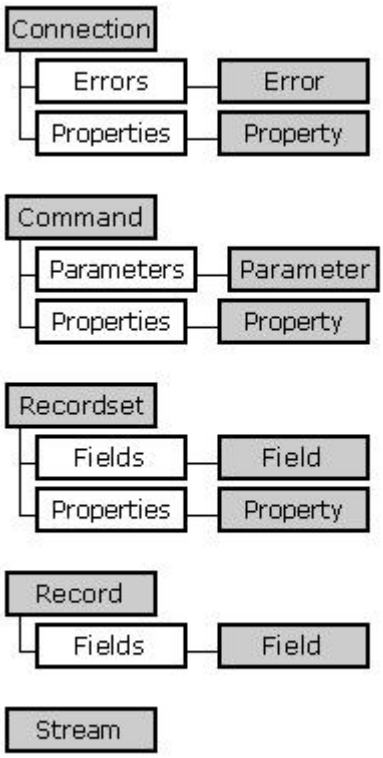
3. Visual Basic 6 kod za rad sa bazom - ADO 2.5

ADO OBJEKTNI MODEL – KRATAK OPIS



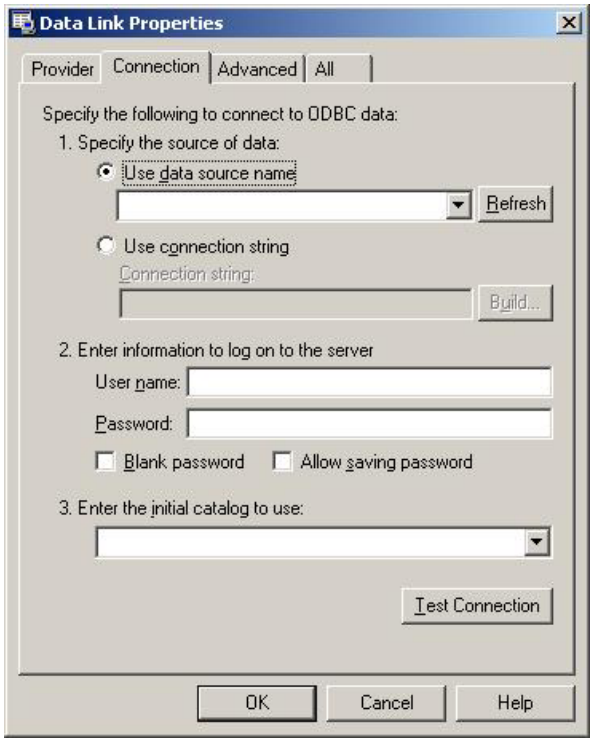
Connection objekat je top level objekat u ADO hijerarhiji. Ovaj objekat predstavlja vezu ka data sourceu i preko njega ide sva komunikacija između VB aplikacije i data sourcea. Važniji metodi su: Open, Close, BeginTrans, CommitTrans i RollbackTrans.
Command objekat predstavlja SQL izraz, stored proceduru ili neku drugu akciju nad podacima. *Parameters* kolekcija prihvata ulazne i izlazne parametre. Obično se koristi za parametrizovane kverije (query) i izvršavanje stored procedura koje vraćaju neki parametar.
Recordset objekat predstavlja skup zapisa koji su rezultat nekog upita i kursor unutar njih. Sadrži kolekciju *Fields* koja sadrži *Field* objekte. *Field* objekat predstavlja jednu kolonu podataka unutar *Recordseta*.
Error objekat sadrži informacije o grešci koju vraća OLE DB provider. Jedan SQL izraz može izazvati više grešaka, pa *Errors* kolekcija može sadržati više *Error* objekata koji su svi posledica istog SQL izraza.

3.1. Povezivanje sa bazom podataka - Connection String

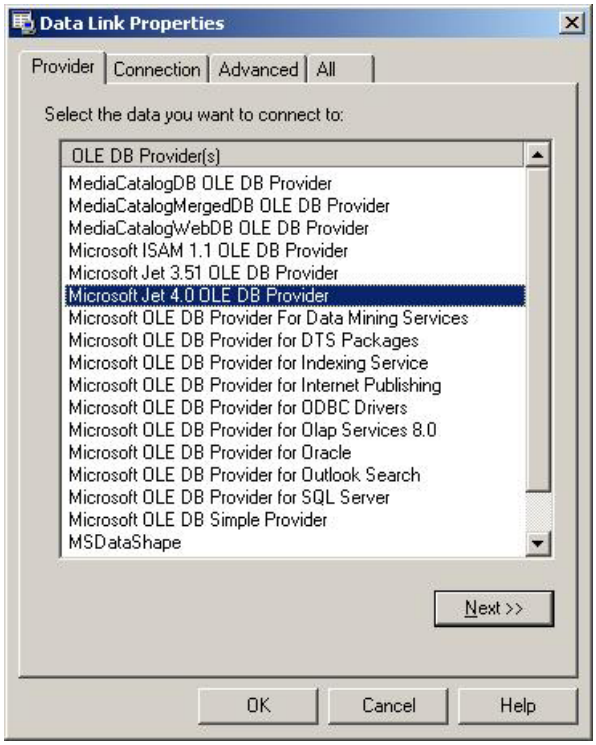
Connection String predstavlja informaciju koja se koristi za uspostavljanje veze sa *data source*om (izvorom podataka, bazom). Ovaj property sadrži niz parova *argument* = *vrednost*, odvojenih znakom ;.

Kako izgleda i kako se formira *Connection String*? Za ovo će poslužiti jedan primer koji je ***degojs*** par puta postovao na VB&ASP forumu :).

- Kreirajte fajl (iz Notepada, ili kako već ko voli) koji se zove, na primer, cs.udl (bitno je da ima ekstenziju UDL).
- Otvorite udl fajl iz Windows Explorera (double click, ili Open, ili ...)
- Trebalo bi da vam se pojavi jedan prozor kao na slici 3.1.1

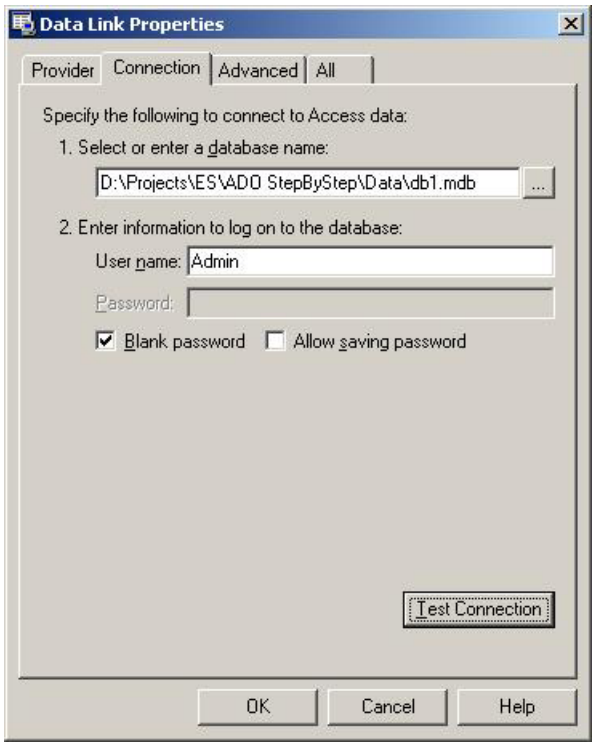


slika 3.1.1



slika 3.1.2

- Vratite se na prvi jezičak (Provider) – slika 3.1.2, i odaberite Providera. Za rad sa Accessom 2000 to će biti *Microsoft Jet 4.0 OLE DB Provider*
- Kliknite na na *Next*
- Na Connection jezičku odaberite putanju do baze (UserName ćemo ostaviti Admin, a password će biti prazan (slika 3.1.3)
- Sada smo sve podesili, i trebalo bi proveriti da li je sve kako treba. Kliknite na dugme test Connection.
- Ukoliko je sve u redu, pojaviće se poruka *Test connection suceeded* (slika 3.1.4)



slika 3.1.3

slika 3.1.4

- Kliknite na OK dugme
- Otvorite fajl u Notepadu
- Fajl bi trebao da izgleda ovako nekako (samo će se path do baze razlikovati):

```
[oledb]
; Everything after this line is an OLE DB initstring
Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=D:\Projects\ES\ADO StepByStep\Data\db1.mdb;Persist
Security Info=False
```

Naš *Connection String* izgleda ovako:

```
Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=D:\Projects\ES\ADO StepByStep\Data\db1.mdb;Persist
Security Info=False
```

OK. Sada imamo *Connection String*. Vreme je da ga iskoristimo.

3.2. Otvaranje konekcije, preuzimanje podataka, zatvaranje konekcije

Uz ovu dokumentaciju ide i jedna VB aplikacija gde je sav kod za ono o čemu se ovde govori. Glavna forma sadrži menije i dugmiće za navigaciju kroz poglavlja ovog dokumenta.

Za početak da objasnimo kako se formira ConnectionString unutar aplikacije. U basMain modulu postoje tri konstante koje se koriste pri formiranju Connection Stringa:

```
'-- connection string constants
Private Const mcstrDSNStart As String =
"Provider=Microsoft.Jet.OLEDB.4.0;Data Source="
Private Const mcstrDSNEnd   As String = ";Persist Security
Info=False"
Private Const mcstrDbPath   As String = "\Data\db1.mdb"
```

Prve dve su fiksne, a treća (`mcstrDbPath`) predstavlja relativnu putanju do baze u odnosu na App.Path. ConnectionString se formira prilikom startovanja sample aplikacije u *Sub Main* (basMain modul)

```
Global gConnectionString    As String
Global gFSO                 As Scripting.FileSystemObject

...

gConnectionString = mcstrDSNStart &
                    gFSO.BuildPath(App.Path, mcstrDbPath) & _
                    mcstrDSNEnd
```

Na ovaj način smo dobili Connection String i koristićemo ga za sve dalje primere. Kod za ovaj primer se nalazi u frmChapter32 formi.

Kako ovo radi? Na formi treba selektovati iz koje tabele/kverija će se selektovati podaci i kliknuti na dugme. Nakon toga će se pojaviti podaci u text boxu sa desne strane.

1. za ovo nam trebaju sledeće promenljive

```
Private m_TableName             As String
...
Dim adoConn As ADODB.Connection ' konekcija
Dim adoRS   As ADODB.Recordset  ' rekordset sa podacima
Dim adoFld  As ADODB.Field      ' polje iz rekordseta
Dim sQry    As String           ' SQL upit
```

Promenljiva `m_TableName` predstavlja ime tabele/kverija odakle će se preuzimati podaci. Ova promenljiva se ažurira čim korisnik klikne na naziv tabele/kverija:

```
Private Sub optTable_Click(Index As Integer)
    ' u captionu se nalazi ime tabele/kverija
    m_TableName = optTable(Index).Caption
End Sub
```

2. kreiranje i otvaranje konekcije

```
    '-- kreiraj i otvori konekciju
    Set adoConn = New ADODB.Connection
    adoConn.Open gConnectionString
```

Konekcija se otvara metodom *Open* Connection objekta, a kao parametar se prosleđuje Connection String. Ako se Connection String ne prosledi kao parametar, onda se uzima vrednost ConnectionString propertya. Znači konekciju smo mogli i ovako da otvorimo:

```
    Set adoConn = New ADODB.Connection
    adoConn.ConnectionString = gConnectionString
    adoConn.Open
```

Pored Connection Stringa, Open metod prihvata i parametre *UserID*, *Password* i *Options*. *UserID* i *Password* se koriste da bi se prosledila informacija o korisniku koji se kači na bazu ukoliko je to potrebno (u ovom slučaju nije). *Options* parametar definiše način izvršavanja Open metoda – sinhrono (default) ili asinhrono. Ako se prosledi **`adAsyncConnect`** onda smo konektovani na bazu kada Connection objekat digne event *ConnectComplete*. U ovom slučaju bi Connection objekat trebalo deklarisati na nivou forme/klase i koristiti WithEvents u deklaraciji.

3. formiranje SQL upita za selekciju podataka

```
Public Const pcstrTableName     As String = "<TableName>"
Public Const pcstrTableQuery    As String = "SELECT * FROM " &
pcstrTableName
```

Ove konstante su definisane u basMain. Znači `pcstrTableQuery` izgleda ovako: `SELECT * FROM <TableName>`. Da bi selektovali podatke iz neke tabele/kverija, potrebno je `"<TableName>"` odnosno konstantu `pcstrTableName` zameniti nazivom željene tabele/kverija:

```
'-- formiraj upit
sQry = Replace(pcstrTableQuery, pcstrTableName, m_TableName)
```

4. kreiranje i otvaranje recordseta

```
    '-- kreiraj i otvori rekordset
    Set adoRS = New ADODB.Recordset
    adoRS.Open sQry, adoConn
```

Recordset objekat se otvara metodom Open. Prvi parametar je *Source* i u ovom slučaju je SQL upit. *Source* može biti Command objekat, poziv stored procedure, SQL upit, ime tabele ili naziv fajla koji sadrži persisted recordset (takav fajl se može dobiti pozivom adoRS.Save). Drugi parametar je aktivna konekcija. To može biti Connection objekat ili string koji predstavlja ConnectionString. Ostali parametri nisu navedeni pa se koriste default vrednosti. Ostali parametri su *CursorType*, *LockType*, *Options*. Ovi parametri zavise od toga šta ćemo raditi sa recordsetom. Za sada se neću baviti njima (u MSDNu ima sve lepo objašnjeno, a i mislim da će se kroz primere videti kad se koji koristi), a možda se kasnije vratim na ovo.

Recordset smo u ovom slučaju mogli otvoriti i ovako:

```
    adoRS.Open m_TableName, adoConn, , , adCmdTable
```

ili ovako:

```
    adoRS.CursorType = adOpenForwardOnly
    adoRS.LockType = adLockReadOnly
    adoRS.Open m_TableName, adoConn, , , adCmdTable
```

Znači *CursorType* i *LockType* parametri se mogu proslediti metodu Open, a mogu se setovati propertyji Recordset objekta.

O prolasku kroz recordset će biti reči u poglavlju 3.4.

3.3. Prikazivanje podataka u DataGrid kontroli

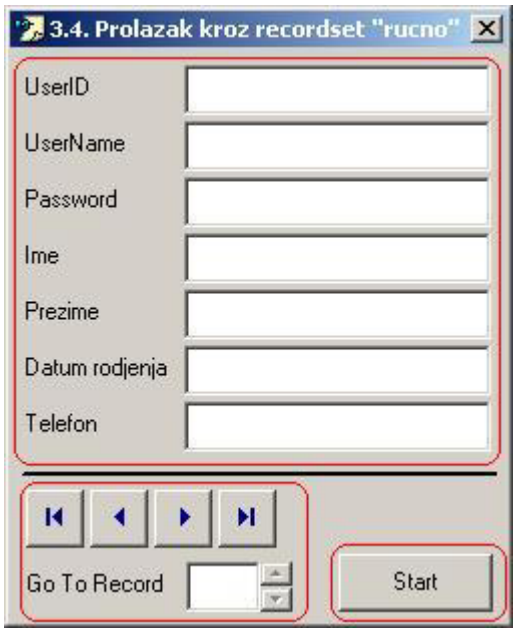
Kod ovog poglavlja se nalazi u formi frmChapter33. Ova forma sadrži jednu grupu option buttona koja služi za biranje tabele iz koje će se prikazivati podaci, DataGrid kontrolu u kojoj će se prikazati podaci i comand button koji otvara konekciju, preuzima podatke i prikazuje ih u gridu. Skoro sav kod je objašnjen u prethodnom poglavlju, a najbitniji deo je

```
'-- prikazi zapise
Set dgrData.DataSource = adoRS
```

DataSource property može biti *ADO Recordset*, kao i klasa ili kontrola koja je definisana kao DataSource (**DataSourceBehavior** property = **vbDataSource**).

3.4. Prolazak kroz recordset ručno

Kod ovog poglavlja se nalazi u formi frmChapter34.



Ova forma sadrži grupu textboxova u kojima će biti prikazani podaci iz tabele *tbl_Korisnici*, dugmiće za navigaciju i dugme *Start* (otvaranje konekcije i rekordseta i preuzimanje podataka).

Textboxovi u kojima se prikazuju podaci imaju podešen *Tag* property u design timeu na naziv polja iz recordseta čija će se vrednost prikazati u tom textboxu.

Metodi *Recordset* objekta koji su bitni za ovo poglavlje su: *Move*, *MoveFirst*, *MovePrevious*, *MoveNext* i *MoveLast*.

Propertyji *Recordset* objekta koji su nam ovde bitni su *BOF*, *EOF* i *AbsolutePosition*.

BOF i *EOF* su tipa Boolean. *BOF* je True ako je pozicija tekućeg zapisa u Recordset objektu pre prvog zapisa (indikator početka). *EOF* je True ako je pozicija tekućeg zapisa u Recordset objektu posle zadnjeg zapisa (indikator kraja). *EOF* i *BOF* su istovremeno True ako ne postoji tekući zapis (current record) tj. ako Recordset ne sadrži nijedan zapis. EOF i BOF služe kao granice za kretanje po Recordset objektu.

AbsolutePosition property omogućava čitanje/setovanje trenutne pozicije unutar *Recordset* objekta. Ovaj property vraća broj u intervalu od 1 do ukupnog broja zapisa u *Recordsetu*, ili jednu od sledeće tri konstante:

- adPosUnknown – ako je *Recordset* prazan, ili ako *provider* ne podržava property
- adPosBOF – ako je *BOF* property True
- adPosEOF – ako je *EOF* True

MoveFirst, *MoveLast*, *MovePrevious* i *MoveNext* metode neću pojašnjavati jer su nazivi metoda deskriptivni.

Move metod omogućava pomeranje za određeni broj zapisa napred ili nazad u odnosu na trenutnu poziciju ili u odnosu na bookmark (Bookmarke pogledati u MSDNu :)).

```
recordset.Move numrecords [,start]
```

`Numrecords` predstavlja broj zapisa za koliko treba izvrsiti pomeranje. Moye biti pozitivan ili negativan i od toga zavisi smer pomeranja. `start` parametar predstavlja zapis u odnosu na koga se vrši pomeranje. Ako se parametar start izostavi, podrazumeva se trenutna pozicija.

Kad se forma otvori, treba kliknuti na dugme Start. Tada se otvara konekcija, otvara se recordset i preuzimaju podaci. Zatim se proverava da li ima zapisa u recordsetu:

```
If Not (m_adoRS.EOF And m_adoRS.BOF) Then
```

Ako recordset nije prazan, podešavaju se parametri UpDown kontrole, koja će se koristiti za demonstraciju *Move* metoda:

```
        udPosition.Min = 1
        udPosition.Max = m_adoRS.RecordCount
```

Na kraju se prikazuje trenutni zapis i trenutna pozicija (`AbsolutePosition` property) u recordsetu:

```
        Call m_DisplayRecord
        txtPos.Text = m_adoRS.AbsolutePosition
```

`m_DisplayRecord` prikazuje trenutni zapis na formi. Zapis se prikazuje u nizu TextBoxova kojima je u design timeu u *Tag* property upisano ime polja koje će se prikazivati u tom textboxu. Kroz niz textboxova se prolazi *For Each ... Next* petljom i prikazuju se vrednosti iz recordseta.

```
    For Each oTextBox In txtField
        If Not IsNull(m_adoRS.Fields(oTextBox.Tag).Value) Then
            oTextBox.Text = m_adoRS.Fields(oTextBox.Tag).Value
        Else
            oTextBox.Text = ""
        End If
    Next
```

Ovo je moglo i drugačije da se implementira (imate u kodu i kako), ali ja radim ovako :).

E, stigosmo do pomeranja. Deo koda za kretanje kroz RS nalazi se u `cmdNavigation_Click`

```
    Select Case Index
        Case 0 ' move first
            m_adoRS.MoveFirst
        Case 1 ' move prev
            If Not m_adoRS.BOF Then m_adoRS.MovePrevious
            If m_adoRS.BOF And m_adoRS.RecordCount > 0 Then
                Beep
                'moved off the end so go back
                m_adoRS.MoveFirst
            End If
        Case 2 ' move next
            If Not m_adoRS.EOF Then m_adoRS.MoveNext
            If m_adoRS.EOF And m_adoRS.RecordCount > 0 Then
                Beep
                'moved off the end so go back
                m_adoRS.MoveLast
            End If
        Case 3  ' move last
            m_adoRS.MoveLast
    End Select
```

Prilikom pomeranja na sledeći ili prethodni zapis, potrebno je proveriti da li smo na kraju (EOF), odnosno početku (BOF).

U `udPosition_Change` može se videti kako se koristi *Move* metod. UpDown kontrola je napunjena brojevima od 1 do broja zapisa. Kad UpDown kontrola ima vrednost *n*, treba se pomeriti na *n*-ti zapis:

```
m_adoRS.Move udPosition.Value - 1, adBookmarkFirst
```

Gore navedeni kod pomera poziciju u recordsetu za n-1 u odnosu na prvi zapis. Ovo se moglo izvesti i ovako:

```
m_adoRS.AbsolutePosition = udPosition.Value
```

3.5. Izmena podataka (UPDATE, DELETE, INSERT)

Podaci se mogu menjati izvršavanjem SQL izraza (UPDATE, DELETE, INSERT) ili u recordsetu metodama Delete, AddNew, Update i UpdateBatch. Pošto se ***degojs*** bavio privm načinom u drugom poglavlju ovog tutoriala, o tome neće biti reči ovde.

Kod koji se odnosi na ovo poglavlje nalazi se u formi frmChapter35. Ova forma liči na formu iz prethodnog poglavlja, s tim što ovde postoje dugmići za dodavanje novog zapisa, brisanje zapisa, izmenu podataka, snimanje izmena, odustajanje od izmena i osvežavanje podataka. Veći deo koda je isti kao u formi frmChapter34, pa će ovde biti objašnjen samo deo koji se odnosi na izmenu podataka.

3.5.1. DODAVANJE NOVOG ZAPISA (ADD)

```
'-- clear fields
Call m_ClearFields(txtField)
'-- otkljucaj kontrole
Call m_LockControls(txtField, False)
With m_adoRS
    If Not (.BOF And .EOF) Then
        '-- zapamti trenutnu poziciju da bi se vratili
        '-- ako korisnik odustane od novog zapisa
        m_Bookmark = .Bookmark
    End If
    .AddNew
    m_Mode = mdAdd
End With
'-- prikazi dugmice
m_ShowButtons cmdAction, "001010"
m_EnableButtons cmdNavigation, "0000"
```

Prvo treba obrisati vrednosti iz textboxova. Da su kontrole bile boundovane ovo bi se automatski desilo. Ako hoćete da radite sa boundovanim kontrolama, VB vam nudi VB Data Form Wizard prilikom dodavanja nove forme, tako da kad odgovorite na sva pitanja wizarda dobijate gotovu formu koja ima sličnu funkcionalnost kao ova naša, samo što su kontrole boundovane. Zatim, treba otključati polja da bi se u njih moglo nešto upisati. Sledeći korak je pamćenje trenutne pozicije u reordsetu, da bi mogli da se vratimo na tu poziciju ako se odustane od dodavanja novog zapisa. I sad ono najvažnije AddNew. Ovaj metod kreira novi zapis u recordsetu i postavlja ga za tekući. Nakon ovoga treba disableovati dugmiće za navigaciju, i prikazati samo Save i Cancel dugmiće. Sada treba uneti vrednosti za novi zapis,

kliknuti na Save da bi se zapis dodao u bazu, ili Cancel da bi se odustalo od dodavanja novog zapisa.

VAŽNO: Ne vrši se provera da li su podaci koji se unose validni (tip, dužina,...). Ukoliko se unese neispravan podataka baza će vratiti grešku.

3.5.2. IZMENA TRENUTNOG ZAPISA (EDIT)

```
'-- otkljucaj kontrole
Call m_LockControls(txtField, False)
m_Mode = mdEdit
'-- prikazi dugmice
m_ShowButtons cmdAction, "001010"
m_EnableButtons cmdNavigation, "0000"
```

Prvo se otključavaju polja da bi se mogle menjati vrednosti, prikazuju se odgovarajući dugmići i sad treba promeniti podatke i zapamtiti ih (Save) ili poništiti izmene (Cancel).

3.5.3. SNIMANJE PROMENA (SAVE)

```
'-- prepisi podatke iz textboxova u RS
Call m_UpdateData
'-- update
m_adoRS.UpdateBatch adAffectAll
'-- pozicioniraj se na novi/izmenjeni zapis
If m_Mode = mdAdd Then
    m_adoRS.MoveLast
    '-- new key is avaliable at this point
    txtField(0).Text = m_adoRS("UserID")
End If
m_Mode = mdNone
'-- prikazi dugmice
m_ShowButtons cmdAction, "110101"
m_EnableButtons cmdNavigation, "1111"
```

Za početak treba prepisati podatke iz textboxova u recordset (da su kontrole boundovane ovo se ne bi radilo). Zatim, ide poziv metoda *UpdateBatch.* Zavisno od parametra, ovaj metod će updateovati trenutni zapis (**adAffectCurrent**) ili sve zapise (**adAffectAll**). Ukoliko je dodat novi zapis, treba se pozicinirati na njega (dodaje se na kraj) i prikazati vrednost ključa. Pošto je ključ Autonumber, njegova vrednost se ne unosi, već baza vodi računa o tome. Nakon poziva metoda UpdateBatch informacija o IDu novog zapisa je dostupna. Na kraju treba prikazati dugmiće. Kad smo već ovde, dugmići su nizovi kontrola i enableuju se i prikazuju metodama `m_EnableButtons` i `m_ShowButtons`. Ove metode prihvataju stringove nula i jednica, pa ako je string npr. 101 to znači da će dugme sa indexom 0 biti

enableovano, dugme sa indexom 1 će biti disbleovano, a dugme sa indexom 2 će biti enableovano, ...

3.5.4. BRISANJE ZAPISA (DELETE)

```
With m_adoRS
    '-- brisi zapis
    .Delete
    '-- pomeri se na sledeci zapis
    .MoveNext
    If .EOF Then .MoveLast
End With
'-- update
m_adoRS.UpdateBatch adAffectAll
```

Metod *Delete* služi za brisanje zapisa. Pošto koristimo *LockType= adLockBatchOptimistic*, zapis će nakon poziva *Delete* metoda biti markiran za brisanje (nema brisanja iz baze), a tek nakon poziva UpdateBatch biće obrisan iz baze. Za više informacija pogledati LockType property u MSDNu.

Brisanje zapisa se moglo obaviti izvršavanjem sledećeg SQL izraza:

```
DELETE FROM tbl_Korisnici WHERE UserID=IDUsera
```

SQL izraz se izvršava pozivanjem *Execute* metode *Connection* objekta (pogledati u MSDNu)

Prilikom brisanja korisnika za kojeg postoje zapisi u tabeli tbl_Korisnici ćete dobiti grešku. Ovo se obično rešava čekiranjem opcije Cascade Delete Related Records na propertyjima veze izmedju dve tabele u samoj bazi. Otvorite bazu, pa Tools/Realtionships, selektujte vezu izmedju dve tabele, desni klik, pa Edit Relationship. Drugi način je da se pre brisanja korisnika, izvrši brisanje svih njegovih textova iz druge tabele.

```
m_adoConn.Execute "DELETE FROM tbl_Tekstovi WHERE UserID=IDUsera "
```

3.5.5. ODUSTAJANJE OD PROMENA (CANCEL)

CancelUpdate metod poništava promene koje su izvršene na tekućem zapisu ili na novom zapisu koji je dodat metodom AddNew. Ukoliko je poništeno dodavanje novog zapisa, potrebno je vratiti se na onaj zapis na kojem je kliknuto na Add.

```

```

```
            '-- cancel update
            Call m_LockControls(txtField, True)
            m_adoRS.CancelUpdate
            '-- pomeri se na zadnju poziciju ako je bilo AddNew
            If m_Bookmark > 0 Then
                m_adoRS.Bookmark = m_Bookmark
            Else
                m_adoRS.MoveFirst
            End If
            m_Mode = mdNone
            '-- prikazi dugmice
            m_ShowButtons cmdAction, "110101"
            m_EnableButtons cmdNavigation, "1111"
```

3.5.6. OSVEŽAVANJE PODATKA (REFRESH)

```
m_adoRS.Requery
```

Requery metod osvežava podatke ponovnim izvršavanjem upita čije podatke recordset sadrži.

3.6. Slanje podataka nazad u bazu (disconnected recordset)

Disconnected recordset je rekordset koji je popunjen podacima iz *data sourcea* i nije nakačen na *data source* (disconnected), tj. ne postoji konekcija ka *data sourceu*. Sve promene koje se vrše su lokalne (ne vide se u bazi). Promene će se reflektovati na bazu ako se *recordset* objekat rekonektuje i ako se pozove *Update* ili *UpdateBatch* metod.

VB kod koji se odnosi na ovo poglavlje nalazi se u formi frmChapter36 i sličan je kodu iz forme frmChapter34. U čemu je razlika?

Kada se preuzmu podaci, *recordset* se diskonektuje i konekcija se zatvara:

```
    '-- diskonektuj RS
    Set m_adoRS.ActiveConnection = Nothing

    '-- zatvori konekciju
    m_adoConn.Close
    Set m_adoConn = Nothing
```

Pošto polja nisu boundovana, pre pomeranja sa tekućeg zapisa potrebno je upamtiti promene. To se radi u funkciji `m_UpdateData`.

```
'--prepisi podatke iz text boxova u RS
For Each oTextBox In txtField
  ' updateuj sve osim UserID (kljuc autonumber)
  If oTextBox.Tag <> "UserID" Then
    m_adoRS.Fields(oTextBox.Tag).Value = oTextBox.Text
  End If
Next
```

Ova funkcija ima opcioni parametar tipa *Boolean* koji govori da li treba rekonektovati *recordset*. Ova funkcija se poziva sa parametrom True (tj. recordset se rekonektuje) samo kad se klikine na dugme *Vrati podatke u bazu*. Znači, kreira se i otvara konekcija, a recordset se rekonektuje:

```
If bReconnectRS Then
    '-- kreiraj i otvori konekciju
    Set m_adoConn = New ADODB.Connection
    m_adoConn.Open gConnectionString

    '-- "rekonektuj" RS
    Set m_adoRS.ActiveConnection = m_adoConn
End If
```

Na kraju treba pozvati UpdateBatch metod:

```
'--updateuj podatke
If bReconnectRS Then m_adoRS.UpdateBatch
```

Nakon ovoga, promene koje su vršene nad podacima videće se i u bazi.

Eto toliko. Ako naidjete na bugove, prijavite ih i srećno programiranje ☺

Pozdrav,
Željko Mladenović, aka mladenovicz