DataAdapter

Čas 4.

Čemu služi?

- Obezbeđuje lak način za rad sa podacima.
- Objedinjuje
  - Insert
  - Delete
  - Update
  - Select
- Ažuriranje podataka u bazi izvodi korisniku na intuitivan način.
- Sadrži:
  - Konekciju
  - Komande

Kreiranje

- string sql = “select * from Customers”
- SqlConnection sqlconn = new ….
- SqlDataAdapter da = new SqlDataAdapter(..,..);

Deo koji definiše select komandu adaptera

- //Kuda sa podacima?
- //Kreiranje DataSet objekta za prihvat podataka
- DataSet ds = new DataSet();
- da.Fill(ds);

- //kako prikazati podatke iz ds?

- Programsko kreiranje
  - Primer: Napraviti DA za prikaz svih podataka iz tabele Odeljenje
  - Dobijene podatke pogledati koristeći debug-er u okviru IDE VS

Svojstva

- **Komande**
- DeleteCommand
- InsertCommand
- SelectCommand
- UpdateCommand

- **...dalje...**

...i dalje o svojstvima

- **TableMappings** – kolekcija koja obezbeđuje relaciju između kolona iz objekta DataSet i izvora podataka. (Kako Fill ume da prebaci podatke iz baze u DS)
- **AcceptChangesDuringFill** – određuje da li se na objektu DataRow poziva AcceptChanges kada se doda u objekat Data Table
- **MissingMappingAction** – definiše akciju koja će se dogoditi kada se ne mogu upariti podaci sa nekom postojećom kolonom ili tablelom
- **MissingSchemaAction** - slično ali za šemu

- U slučaju greške moguće akcije za MissingMappingAction su:
  - **Error**
  - **Ignore**
  - **Passthrough** – kolona ili tabela koja se ne pronađe dodaje se u DataSet korišćenjem njenog imena u izvoru podataka

- Svojstva za MissingSchemaAction su:
  - **Add** dodaje potrebne kolone
  - **AddWithKey** dodaje potrebne kolone i ograničenja primarnog ključa
  - **Error** Ispaljuje izuzetak
  - **Ignore** Ignoriše dodate kolone

Primer

- Koristeći VS:
  - Pokazati komande
  - Pokazati promene komandi

TableMappings:

1. Kolekcija
2. Relacije između izvora podataka sa jedne strane i skladišta podataka (DataSet-a) sa druge strane.
3. Vezuje se ime kolone u tabeli u bazi sa imenom kolone u tabeli u DataSet-u

Metode objekta DataAdapter

1. Fill
2. Update

Fill metoda

- Popunjava DataSet podacima sa prethodim brisanjem postojećih ili ne.
  - To se podešava u određenim svojstvima – pronađite kako

Update

- Promene koje su učinjene na podacima se prebacuju u bazu.
- Kako?
  - Pogledajte kako glasi sql upit komande Update.

CommandBuilder

- SqlDataAdapter da = new SqlDataAdapter();
- SqlCommandBuilder bu = new SqlCommandBuilder(da);
- da.DeleteCommand = bu.GetDeleteCommand();
- da.InsertCommand = bu.GetInsertCommand();
- .....
- da.Update();

- Napomena: Samo za tabele sa primarnim ključom i već pripremljenim svojstvom SelectCommand

Moguće je dodati i proizvoljne metode sa specifičnim upitima!

Primer: Pokazati dodavanje proizvolje metode koristeći VS.

Događaji

- Događaji greške
- OnRowUpdating
  - Odmah posle trenutka kada metod Update postavi vrednosti parametara komande koja treba da se izvrši ali pre samog izvršavanja.
- Argument uz ovaj događaj
  - OleDbRowUpdatingEventArgs
  - SqlRowUpdatingEventArgs
    - Svojstva ovog objekta su:
      - Command - komanda za podatke koja treba da se izvrši
      - Errors – greške koje .NET generiše
      - Row – objekat DataReader koji treba da se ažurira
      - StatementType – *Select,Insert,Update,Delete*
      - TableMapping – Objekat DataTableMapping koji se koristi za ažuriranje

Primer:

- Kreiraćemo metodu koja je rukovaoc tj. koja će odgovarati na događaj OnRowUpdating objekta DataAdapter sa ispisom svih svojstava.

OnRowUpdated

- Kada metod Update izvrši odgovarajuću komandu na izvoru podataka
  - Svojstva argumenta koji ide uz ovoj događaj pogledaćemo na primeru.

DataSet

Čas 5.

Šta je i čemu služi?

- DataSet je jedan od najbolje osmišljenih i najviše korišćenih objekata ali je i jedna od važnijih karakteristika Microsoftove .NET platforme.
- Sličan je tradicionalnom ADO recordsetu, ali i sa bitnim razlikama.

- **1)DataSet može da čuva rezultate više različitih SQL upita.**
- **2)Možete koristiti ovaj objekat nezavino od konekcije.**
- **3)Može se kreirati iz XML fajla .**
- **4)Možete kreirati XML fajl iz DataSeta.**

Od čega se sastoji?

- Kolekcija tabela
  - Tabela
    - Kolekcija redova
      - Red
    - Kolekcija kolona
      - Kolona
    - Ograničenja
      - Ograničenje
- Veze
  - Veza

Kako se koristi?

- Pretpostavimo da imamo konekciju do SQL servera, kreirajmo data adapter na sledeći način:

```
sqlDataAdapter1.SelectCommand.CommandText="SELECT * FROM radnik“
DataSet ds= new DataSet();
sqlDataAdapter1.Fill( ds ) ;
DataTable dt = ds.Tables[0];
DataColumn dc       = dt.Columns[2];
                    = dt.Columns[“naziv”];
DataRow dr = dt.Rows[2];
```

- Rezultujući DataSet se sastoji od Tablela koje su označene indeksima počev od 0.
- *listBox1.DataSource=ds.Tables[0] ;*

Vrste DataSet objekata

- **Tipizirani**
- **Netipizirani**

Neka svojstva DataSet-a

- ***Tables***
- *Relations*
- *HasErrors* – označava da li neki od objekata *DataRow* sadrži grešku
- *EnforceConstraints* – određuje da li se prilikom izmena poštuju pravila za ograničenja
- *DataSetName*
- *CaseSensitive* – da li su poređenja osetljiva na velika slova

Metode objekta DataSet

- *Clear* – briše sve tabele
- *Clone* – kopira strukturu DataSet-a
- *Copy* – kopira i strukturu i podatke
- *HasChanges* – da li u objektu DataSet postoje izmene koje čekaju.

Filtriranje i uređivanje u okviru DataSeta

- 1. DataSet objekat napuniti pomoću DA podacima iz tabela *customers* odnosno *products.*
- *string SQL = "select * from customers; select * from products where UnitPrice <10";*
- *da.SelectComand = new SqlCommand(SQL, conn);*
- *da.Fill(ds);*
- *da.Fill(ds, "Customers");*
- // proveri kreirane tabele u ds i njihova imena u nekoliko varijanti popunjavanja

Filtriranje, sortiranje

- DataTableCollection dtc = ds.Tables;
- dtc["Customers"].Select("*xxxx*", "*yyyy*");

- pr: xxxx : Country = 'Germany'
- pr: yyyy: CompanyName ASC

Dodavanje reda

- DataRow newrow= dt.NewRow();
- newrow[“”] = ;
- ….
- dt.Rows.Add(newrow);

Prenošenje izmena

- UpdateCommand
- InsertCommand
- DeleteCommand

CommandBuilder

- SqlDataAdapter da = new SqlDataAdapter();
- SqlCommandBuilder bu = new SqlCommandBuilder(da);
- da.DeleteCommand = bu.GetDeleteCommand();
- da.InsertCommand = bu.GetInsertCommand();
- .....
- da.Update();

- Napomena: Samo za tabele sa primarnim ključom i već pripremljenim svojstvom SelectCommand

Upravljanje konfliktima

- Pesimistička
- Optimistička

Rad sa fajlovima

- Podaci se mogu čuvati u fajlovima koristeći jaku podršku za XML.
- ***ds.write(filename) ;***
- Ako želite da pročitate zapisane podatke koristite
  - ***ReadXml***
- dsPreduzece.WriteXmlSchema("C:\\dspreduzeceSema.xml");
- dsPreduzece.WriteXml("C:\\dspreduzece.xml", XmlWriteMode.WriteSchema);

Uraditi primer i pogledati dobijeni fajl.

Primer

- 1. U kodu kreiraj adapter za preuzimanje podataka iz tabele Odeljenje
- 2. U kodu kreiraj ostale adaptera za ažuriranje podataka tabele Odeljenje
- 3. Koristeći 2. uradi proveru
- 4. Obezbedi filtriranje po imenu
  - Pri filtriranju kreirati pomocnu tabelu na osnovu niza redova dobijenih primenom select naredbe objekta DataTable

Primer: delovi koda

```
conn = new OleDbConnection(@„...");
da = new OleDbDataAdapter("select * from odeljenje", conn);
bu = new OleDbCommandBuilder(da);
----
da.Fill(ds);
-----
 private void btnKreirajAdapter_Click(object sender, EventArgs e
 {
    da.UpdateCommand = bu.GetUpdateCommand();
    da.DeleteCommand = bu.GetDeleteCommand();
    da.InsertCommand = bu.GetInsertCommand();
 }
-----
```

Primer:formiranje tabele filtriranih redova

```
DataRow[] fltRedovi = ds.Tables[0].Select("imeod like'" + textBox1.Tex

DataTable dtF = new DataTable();

DataColumn colIme = new DataColumn("imeOdeljenja", Type.GetType("Syste
dtF.Columns.Add(colIme);
DataColumn colMesto = new DataColumn("Mesto", Type.GetType("System.Str
dtF.Columns.Add(colMesto);

foreach(DataRow red in fltRedovi)
{
     DataRow dr = dtF.NewRow();
     dr["imeOdeljenja"] = red["imeod"];
     dr["Mesto"] = red["mesto"];
     dtF.Rows.Add(dr);
}
dgvF.DataSource = dtF;
```