DataAdapter i DataSet

DataAdapter?

- Obezbeđuje lakši način za rad sa podacima. Objedinjuje sve 4 vrste sql komandi:
  - Insert, Delete, Update, Select koristeći 4 Command objekta.
- Ažuriranje podataka u bazi vrši na intuitivan način.
- Sadrži:
  - Jedan objekat `Connection`
  - Četri objekta Command
  - Svojstva, Metode, Događaje

Svojstva

- **`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. U slučaju greške moguće akcije su:
  - **`Error`**
  - **`Ignore`**
  - **`Passthrough`** – kolona ili tabela koja se ne pronađe dodaje se u DataSet korišćenjem njenog imena u izvoru podataka
- **`MissingSchemaAction`** - slično ali za šemu. Svojstva su:
  - **`Add`** - dodaje potrebne kolone
  - **`AddWithKey`** - dodaje potrebne kolone i ograjčenja primarnog ključa
  - **`Error`** - daje izuzetak
  - **`Ignore`** - Ignoriše dodate kolone

Metode

Fill

- Popunjava objekat tipa DataSet podacima. Moguće je prethodno brisanje ili čuvanje postojećih.
  - To se podešava u određenim svojstvima – pronađite kako

Update

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

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

Kreiranje

- Zasniva se na upitu tipa Select.
- Ukoliko se bazira na tabeli sa primarnim ključem onda može da istovremeno formira Insert, Update i Delete komande. Ako nema to onda formira samo Select komandu.

```
string sql = “select * from Radnik”;
OleDbConnection sqlconn = new ….
OleDbDataAdapter da = new OleDbDataAdapter(..,..);
```

```
//Kuda sa podacima?
//Obično se koristi DataSet objekat za prihvat podataka
DataSet ds = new DataSet();
da.Fill(ds);
```

DataSet

Sadrži kolekciju tabela: DataTable.

Svaki DataTable sadrži kolekciju kolona – DataColumn, kao i kolekciju redova DataRow. Organizacija ovih kolekcija odgovara tabelarnoj organizaciji podataka.

Metode

- `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 prilikom popunjavanja objekta DataSeta

- DataSet objekat napuniti pomoću DA podacima iz tabele ili više tabela sa ili bez uslova filtriranja.
- `string SQL = “select * from Radnik where ime like „.“;`
- `da.SelectComand = new OleDbCommand(SQL, conn);`
- `da.Fill(ds);`
- `da.Fill(ds, “Radnik”);`
- proveriti kreirane tabele u ds i njihova imena u nekoliko varijanti popunjavanja.
- Prilikom popunjavanja moguće je uraditi filtriranje primenom sql upita, na primer:
- `SQL = “select * from Radnik where ime like „. . .“;`

Dodavanje reda

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

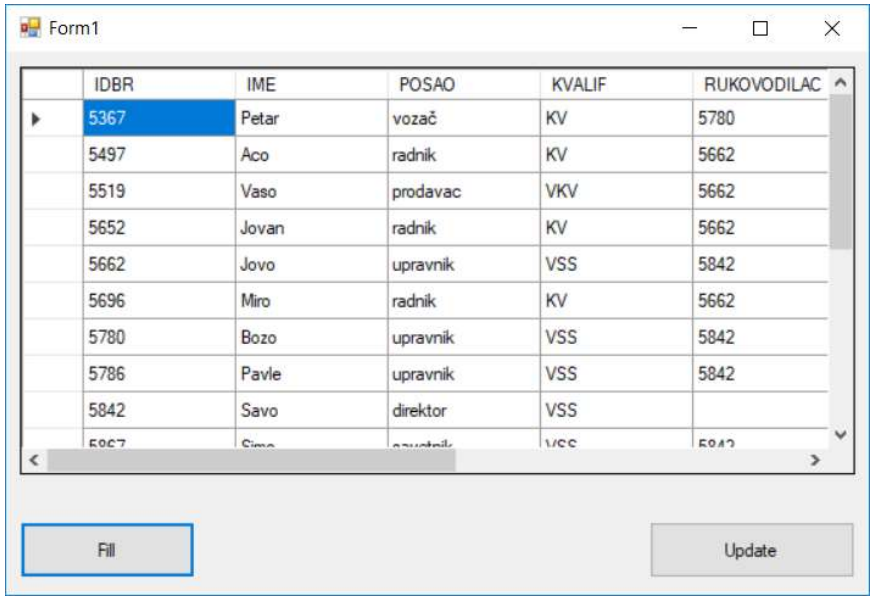
Filtriranje, sortiranje redova u DataSet-u

- `DataTableCollection dtc = ds.Tables;`
- `dtc[“Customers”].Select(“fltStr”, “sortStr”);`

- `pr: fltStr: Country = ‘Germany’`
- `pr: sortStr: CompanyName ASC`

Primer

- Napraviti aplikaciju za rad sa podacima tabele, kao na slici. Obezbediti:
  - prikaz podataka
  - Ažuriranje promena
  - Filtriranje po imenu.



Dodavanje objekata DataSet i DataAdapter

```
DataSet ds;
OleDbDataAdapter da;
public Form1()
{
    InitializeComponent();

    ds = new DataSet();
    da = new OleDbDataAdapter("select * from Radnik", @"kon.string");
}
```

```
private void btnFill_Click(object sender, EventArgs e)
{
    da.Fill(ds);
    dataGridView1.DataSource = ds.Tables[0];
}
```

Kopletiranje adaptera

- Ukoliko kreirani adapter nema formirane komande za željene operacija one se mogu kreirati iz koda primenom objekta CommandBuilder. U našem slučaju to bi bilo:

```
OleDbCommandBuilder cmdBuild = new OleDbCommandBuilder(da);
da.InsertCommand = cmdBuild.GetInsertCommand();
da.UpdateCommand = cmdBuild.GetUpdateCommand();
da.DeleteCommand = cmdBuild.GetDeleteCommand();
```

Snimanje promena

- Snimanje svih vrsta promena izvodi se preko jedne metode adaptera.

```
private void btnUpdate_Click(object sender, EventArgs e)
{
    da.Update(ds.Tables[0]);
}
```

- Zapazite da je metoda Fill kreirala prvu tabelu u DataSet objektu. Pri tome su kreirane odgovarajuće kolone i dodati su redovi.
- Takav objekat se zatim provezuje sa DataGridView kontrolom i prikazuju podaci.

Izdvajanje redova iz DataSet objekta - pomoću Select naredbe

```
private void btnFilter1_Click(object sender, EventArgs e)
{
    DataTable dt = ds.Tables[0];
    DataTable dtRez = dt.Clone();
    DataRow[] rows = dt.Select("ime like '" + txtIme.Text + "%'");
    foreach (DataRow row in rows)
    {
        DataRow newrow = dtRez.NewRow();
        for (int i = 0; i < dt.Columns.Count; i++)
            newrow[i] = row[i];
        dtRez.Rows.Add(newrow);
    }

    dataGridView1.DataSource = dtRez;
}
```

Izdvajanje redova iz DataSet objekta - pomoću Linq-a

```
private void btnFilter2_Click(object sender, EventArgs e)
{
    IEnumerable<DataRow> rows = ds.Tables[0].Rows.Cast<DataRow>();
    IEnumerable<DataRow> rez = rows.Where(x =>
((string)x["ime"]).StartsWith(txtIme.Text)).Select(x=>x);

    DataTable dtRez = new DataTable();
    if(rez.Count() > 0)
        dtRez = rez.CopyToDataTable();

    dataGridView1.DataSource = dtRez;
}
```

- Rad sa više formi
  - Obično se može koristiti isti DS u radu sa više formi.
  - U tom slučaju se preko formi i kontrola obezbeđuje različit prikaz za iste podatke i automatizovano osvežavanje svih izmena u prikazu.
- U primeru koji sledi prikazaćemo dodavanje novog reda u posebnoj formi.

- Na novoj formi koristićemo isti DataSet objekat kao i na prvoj formi aplikacije.
  - Zbog toga ćemo preneti referencu iz prve forme preko novog konstruktora nove forme

```
DataTable dt;
OleDbDataAdapter da;
public FrmNoviRadnik()
{
    InitializeComponent();
}
public FrmNoviRadnik(DataTable _dt, OleDbDataAdapter _da) : this()
{
    dt = _dt;
    da = _da;
}
```

Dodavanje novog reda

```
DataRow noviRed = dt.NewRow();
noviRed["IDBR"] = int.Parse(txtIDBR.Text);
noviRed["ime"] = txtIme.Text;
dt.Rows.Add(noviRed);
```

Snimanje promena

```
private void btnSacuvaj_Click(object sender, EventArgs e)
{
    try
    {
        DataRow noviRed = dt.NewRow();
        noviRed["IDBR"] = int.Parse(txtIDBR.Text);
        noviRed["ime"] = txtIme.Text;
        dt.Rows.Add(noviRed);

        da.Update(dt);
    }
    catch(Exception ex)
    {
        MessageBox.Show(ex.Message);
    }

}
```

Zatvaranje forme bez snimanja

```
private void btnCancel_Click(object sender, EventArgs e)
{
    // Ostao neresen problem poništavanja promena u DataSet-u
    this.Close();
}
```

Povezivanje kontrola - 1

- ComboBox i ListBox prikazuju 1 kololonu tabela. Eventualno mogu povezati i svojstvom ValueMemember još jednu kolonu sa podacima.

```
// cb i lb
listBox1.DataSource = dt1;
listBox1.DisplayMember = "ime";

comboBox1.DataSource = dt1;
comboBox1.DisplayMember = "ime";
```

Povezivanje kontrola - 2

- Povezivanje prostih kontrola koje pokazuju jednu vrednost ide preko nekog svojstva kontrole. Za ovo se koristi kolekcija povezivanja. Za jednu kontrolu se može uraditi povezivanje nekoliko različitih podataka iz tabele.

```
// povezivanje sa jednostavnim kontrolama
txtIme.DataBindings.Add("Text", dt1, "ime");
dtpDatumZap.DataBindings.Add("Value", dt1, "datzap");
progressBar1.DataBindings.Add("Value", dt1, "plata");
```