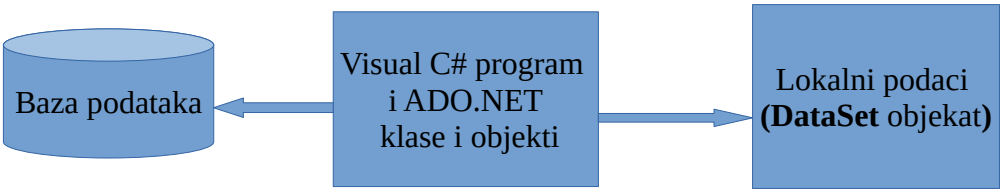
Rad u diskonektovanom okruženju



ADO.NET je Biblioteka klasa za rad sa bazama podataka.
Kada se govori o radu u **diskonektovanom okruženju**, misli se na sledeće: veza sa bazom podataka se koristi samo kod učitavanja podataka iz baze ili kod ažuriranja podataka u bazi. Sve operacije mogu da se obavljaju bez odlaska u izvornu bazu podataka, koristeći lokalnu kopiju. Izmene koje se izvrše u lokalu, prenose se na izvornu bazu tek onda kada se konkretno pokrene **update,** tj ažuriranje u izvornoj bazi podataka.

U slučaju diskonektovanog režima rada, ADO.NET vrši uvoz podataka iz izvorne baze u svoj sopstveni **database manager (DBMS)**, koji je predstavljen **DataSet** objektom (klasom). Objekat klase **DataSet** čuva sve promene nad podacima i kada izvorma baza potadaka treba da se ažurira, program vrši izmene u izvornoj bazi.

ADO.NET KLASE

Tačna kombinacija i tipovi uključenih **ADO.NET** objekata zavisi od izabranog tipa veze (**connection)** između aplikacije i baze.

Na primer:

OLEDB: OleDbCommand, OleDbConnection, OleDataAdapter
MySQL: MySqlCommand, MySqlConnection, MySqlDataAdapter

a objekat klase **DataSet** je uvek isti, bez obzira na tip veze sa bazom podataka.

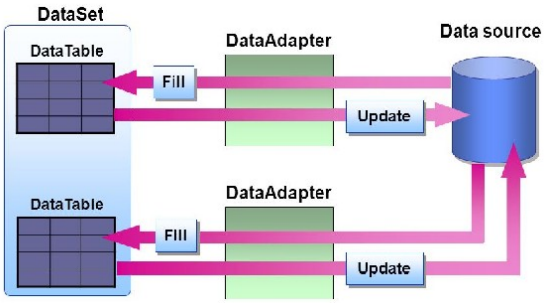
Ukratko:
DataSet je u stvari lokalni DBMS koji je u okviru ADO.NET. Služi za rad sa tabelama, a same tabele su u obliku **DataTable** objekata.

> **DataTable** predstavlja skup zapisa. Najčešće su ova tabele dobijene pretragom nad tabelama u izvornoj bazi podataka.
> **DataView** služi za prikaz podataka iz neke tabele. Obično u svrhu sortiranja ili filtriranja zapisa u nekoj tabeli.

Connection upravlja vezom sa izvornom bazom podataka, npr pomoću metoda Open i Close
Command predstavlja SQL instrukcije (select, update, delete, insert)
DataAdapter služi za import/eksport između DataSet i izvorne baze podataka.

Klasa **DataAdapter** predstavlja most između podataka i objekta DataSet. Objekat DataSet se koristi za čuvanje lokalne kopije podataka baze. Pri čitanju podataka iz baze, objekat DataAdapter prima podatke od objekta Connection i prosleđuje ih objektu DataSet, a promene nad podacima iz objekta DataSet prosleđuje nazad do objekta Connection kako bi se podaci ažurirali u samom izvoru podataka.

Objekat klase DataSet sadrži jednu ili više lokalnih kopija tabela iz baze podataka. Za svaku tabelu u DataSet-u se kreira po jedan objekat klase DataAdapter koji služi kao most između izvora podataka i njegove lokalne kopije koja se nalazi unutar objekta DataSet.
Objekat klase DataAdapter predstavlja skup SQL komandi i konekcije na bazu koja se koristi da se napuni objekat DataSet podacima iz baze i da se promenama izvršenim na objektu DataSet ažurira baza podataka.



Izvor adaptera zavisi od baze sa kojom se radi, odnosno od snabdevača podataka koji se koristi (OleDbDataAdapter, SqlDataAdapter, MySqlDataAdapter, …)

Struktura DataAdaptera

Objekat DataAdapter sadrži SelectCommand, InsertCommand, UpdateCommand i DeleteCommand

Podrazumevano je da su identični nazivi kolona u bazi podataka i odgovarajućem DataTable objektu koji se nalazi u DataSet-u.
DataAdapter koristi Command objekat da komunicira sa bazom.

- SelectCommand predstavlja komandu kojom se čitaju podaci iz baze.
- Insert, Update i Delete se koriste da bi se promene nad podacima u DataSet-u iskoriste za ažuriranje podataka u bazi

Pri kreiranju DataAdapter objekta mora se specificirati SELECT komanda.

Svojstva i metode DataAdaptera
Svojstvo **SelectCommand** objekta klase **DataAdapter vraća **Command objekat koji predstavlja SQL komandu za čitanje zapisa iz baze podataka.
Slično tome, svojstva **InsertCommand**, **UpdateCommand** i **DeleteCommand** vraćaju odgovarajuće **Command objekte koji predstavljaju INSERT, UPDATE i DELETE komandu.
Metoda **Fill** objekta **DataAdapter klase se koristi za punjenje tabele unutar DataSet objekta podacima iz baze.
Metoda **Update** se koristi za ažuriranje baze podataka promenama u DataSet objektu.

Kreiranje objekta DataAdapter

Kreira se pozivom jednog od konstruktora:

```
*DataAdapter(){ }
*DataAdapter(*Command mySqlCommand){ }
*DataAdapter(string selectCommandString, *Connection mySqlConnection){ }
*SqlDataAdapter(string selectCommandString, string connectionString){}

//KreiranjeAdaptera1

//kreira se konekcija za svaki adapter

*DataAdapter da1 = new *DataAdapter(sqlSelectUpit, konekcioniString);
*DataAdapter da2 = new *DataAdapter(sqlSelectUpit, mojaSqlKonekcija);
*DataAdapter da3 = new *DataAdapter(komanda);
*DataAdapter da4 = new *DataAdapter() ;

da4.SelectCommand = komanda;
```

(umesto * upotrebite nastavak za verziju koju koristite)
Ovde je ilustrovana upotreba različitih konstruktora ove klase:

U prvom slučaju, konstruktoru klase *DataAdapter se prosleđuje SELECT upit i odgovarajući konekcioni string. U drugom slučaju, konstruktoru se prosleđuje SELECT upit i odgovarajući *Connection objekat. U trećem slučaju je najpre kreiran *Command objekat koji predstavlja SELECT naredbu, a zatim je on prosleđen kao parametar konstruktoru. U četvrtom slučaju se poziva podrazumevani konstruktor, a odgovarajućem *DataAdapter objektu se preko svojstva SelectCommand dodeljuje odgovarajući *Command objekat koji predstavlja SELECT komandu.

Punjenje DataSet-a

U sledećem delu programskog koda prikazano je kreiranje DataAdapter objekta korišćenjem SELECT komande kao i njegova upotreba za punjenje odgovarajućeg DataSet-a.
Da bi se pozvala Fill metoda DataAdapter-a najpre se mora otvoriti konekcija. Posle poziva Fill metode konekcija se zatvara. Prvi parametar Fill metode je naziv DataSet-a a drugi naziv tabele koju treba kreirati u DataSet-u.

```
*Command mojaKomanda= new *Command();
mojaKomanda.Connection= mojaKonekcija();

mojaKomanda.CommandText="SELECT TOP 5 ProductID, productName, UnitPrice FROM Products ORDER BY ProductID";

*DataAdapter mojDataAdapter= new *DataAdapter();
mojDataAdapter.SelectCommand= mojaKomanda;

DataSet mojDataSet= new DataSet();

mojaKonekcija.Open();

int brojVrsta= mojDataAdapter.Fill(mojDataSet, “Products“);

mojaKonekcija.Close();
```

DataSet

DataSet je u stvari memorijska predstava podataka i u sebi uključuje tabele, relacije između tabela i ograničenja.

- Objekat DataTable se koristi da bi se predstavila tabela u DataSet-u.
- Konekcija sa bazom podataka nije neophodna da bi se manipulisalo sa podacima u DataSet-u.
- Podaci u DataSet-u se čuvaju na sličan način kao što se čuvaju u relacionoj bazi podataka.
- Podaci iz DataSet-a se mogu prikazati u XML formatu.

Svojstva objekata klase DataSet
Klasa DataSet ima

- svojstvo Tables koje daje kolekciju DataTable objekata u DataSet-u.
- Svojstvo Relations koje daje kolekciju objekata DataRelation koji se koristi za opis veza između tabela u DataSet-u.

Objekat klase DataTable ima sledeće kolekcije:

- Columns (kolekcija objekata DataColumn)
- Rows (kolekcija objekata DataRow)
- Constraints (kolekcija objekata Constraint)
- ChildRelations (kolekcija objekta DataRelation)

Kreiranje objekta klase DataSet

```
DataSet ds = new DataSet(“imeBaze”);
DataTable dt = new DataTable(“mojaTabela”);

ds.Tables.Add(dt);

DataColumn jednaKolona = new DataColumn(“nazivKolone”, typeof(tipPodatakaKolone));

dt.Columns.Add(jednaKolona);

jednaKolona = dt.Columns.Add(“nazivKolone”, typeof(tipPodatakaKolone));

jednaKolona.AllowDBNull = false;
```

Ako se konstruktoru ne prosledi ime DataSet-a, onda će mu biti dodeljeno podrazumevano ime NewDataSet. Zatim se instancira DataTable objekat i dodaje u kolekciju Tables DataSet objekta. Kada se prvi put kreira DataTable objekat, on nema šemu. Zato se moraju kreirati objekti klase DataColumns i dodati u Columns kolekciju objekta klase DataTable.

- Objekat DataColumn se može kreirati pozivom DataColumn konstruktora.
- Drugi način na koji se može kreirati objekat DataColumn je direktno dodavanje naziva kolone i tipa podataka za kolonu u Columns kolekciju objekta klase DataTable

Ako vrednosti NULL nisu dozvoljene za kolonu, onda se to specificira korišćenjem svojstva AllowDBNull i njegovim postavljanjem na false.
Ako želimo da vrednosti u koloni imaju jedinstvene vrednosti, onda se to definiše korišćenjem svojstva Unique.

DataTable

Za svaki objekat DataTable koji se nalazi unutar DataSet-a potrebno je definisati primarni ključ.

Objekat DataTable sadrži:

- kolekciju DataColumnCollection koja sadrži objekte klase DataColumn koji predstavljaju odgovarajuće kolone tabele.
- Kolekciju DataRowCollection koja sadrži objekte klase DataRow (predstavljaju redove tabele),
- kolekciju ConstraintCollection koja sadrži objekte klase Constraint (predstavljaju ograničenja definisana nad tabelom)

Punjenje objekta DataTable korišćenjem DataAdapter-a
Objekat DataTable koji se nalazi unutar objekta DataSet se može napuniti korišćenjem Fill metode DataAdapter objekta. Metoda Fill DataAdapter-a izvršava njegovu SELECT komandu.
U sledećem kodu se pomoću Fill metode u DataSet-a kreira tabela Kupci i puni podacima iz baze podataka:

```
*DataAdapter.SelectCommand = mojaSelectKomanda;
DataSet mojDataSet = new DataSet();

*Connection.Open();

int brojRedova = *DataAdapter.Fill(mojDataSet, “Kupci”);

*Connection.Close();
```

DataGridView

DataGridView je kontrola koja se koristi za prikaz tabelarnih podataka. Može da se poveže sa jednom tabelom iz baze ili sa objektom koji je izveden iz tabele.
Korišćenjem svojstva DataSource povezuje se sa objektom klase DataTable ili objektom koji je izveden iz klase DataTable.
Ako se izvrši povezivanje sa objektom klase DataSet, mora se specificirati i njeno svojstvo DataMember kojim se naznačava sa kojom tabelom se vrši povezivanje.
DataGridView kreira po jednu kolonu za svaku kolonu iz izvora podataka. Header kolone kreira na osnovu naziva odgovarajućeg atributa iz tabele.
Dozvoljava različite tipove selekcije: može se selektovati jedna ili više vrsta, jedna ili više kolona, jedna ili više ćelija, ili cela tabela (klikom na ćeliju u gornjem levom uglu).

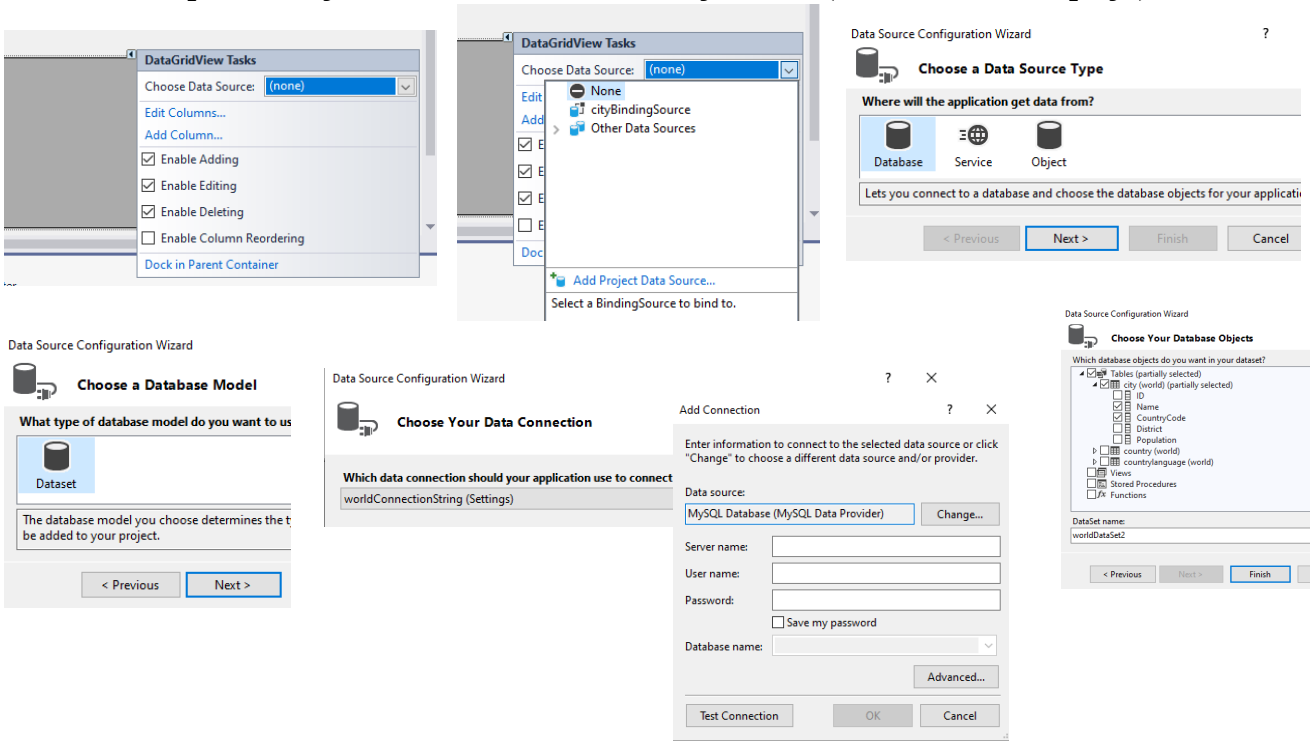
Osobine DataGridView
Korišćenjem tastera Tab prelazi se iz jedne u drugu ćeliju tabele.
Ima automatski tooltip kojim se prikazuje ceo sadržaj ćelije kada se pokazivač miša nalazi iznad nje.
Podržava automatsko sortiranje, klikom na header kolone.
Dvostrukim klikom između header-a kolona vrši se automatsko podešavanje širine kolone.
Omogućava unos sadržaja u ćeliju nakon dvostrukog klika na ćeliju.

Povezivanje DataGridView kontrole sa tabelom iz koda:

```
dataGridView1.DataSource = nds.Proizvodi;
dataGridView1.DataSource = nds;
```

Pomoću svojstva DataSource specificira se tabela iz tipiziranog DataSet objekta sa kojim se povezuje DataGridView kontrola.
Drugi način je da se kao DataSource svojstvo specificira DataSet objekat, ali se u ovom slučaju mora definisati DataMember svojstvo, odnosno tabela iz DataSet-a sa kojom se vrši povezivanje.

Kako se vrši povezivanje DataGridKontrole u dizajn modu (ako IDE ima tu opciju):



Klikom na strelicu u gornjem desnom uglu DataGridView kontrole otvara se prozor DataGridView Tasks. U ovom prozoru se klikom na ComboBox kontrolu Choose Data Source specificira tabela iz DataSet-a koju treba prikazati unutar DataGridView kontrole.
Ovim postupkom se automatski kreira odgovarajući BindingSource objekat kao i punjenje tabele iz DataSet-a podacima iz baze podataka primenom DataAdapter klase.
Kod koji puni tabelu unutar DataSet-a se dodaje Load proceduri forme.