

# **Povezivanje podataka i složene zaslonske maske**

---

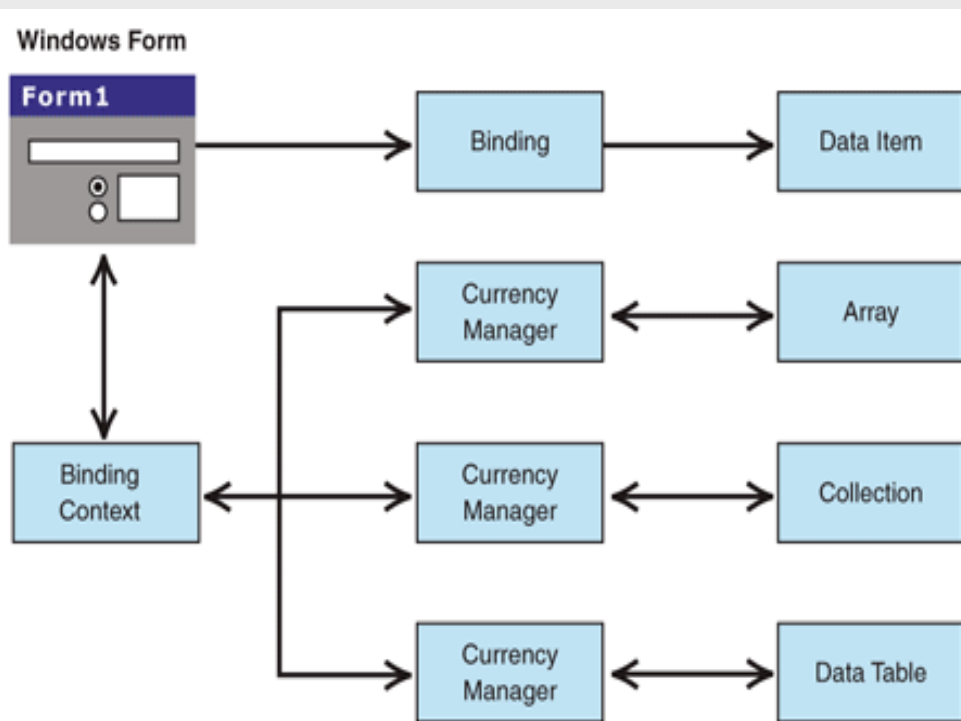
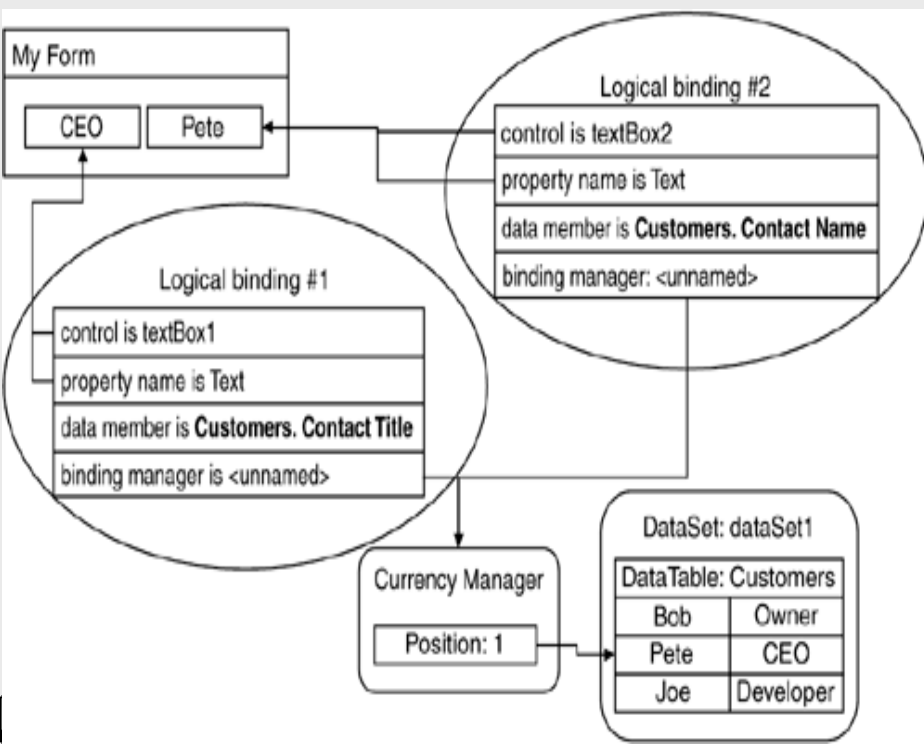
**2014/15.06**

# Povezivanje podataka s kontrolama na formi

## ❑ **Data Binding** - Mehanizam vezanja (povezivanja, privezivanja) elemenata grafičkog sučelja na podatke

- povezuje podatke (izvor podataka) sa svojstvima kontrola, najčešće s njihovim vrijednostima, a općenito s bilo kojim svojstvom
  - npr. `ListBox.DataMember` ili `TextBox.Text`
  - npr. `ForegroundColor`, `Font`

## ❑ Povezivanje se ostvaruje putem suradnje nekoliko vrsta objekata



# Objekti koji sudjeluju u povezivanju

- ❑ **Izvor podataka:** objekt s javnim svojstvima, niz, kolekcija koja implementira sučelje *IList*, složeniji tip podataka (npr. *DataSet* , *DataTable*).
- ❑ ***CurrencyManager***
  - vodi evidenciju o trenutnoj poziciji unutar izvora podataka (izvor ne zna koji se element trenutno prikazuje).
  - Za svaki izvor podataka postoji zasebna instanca, a za više kontrola iste forme koje se povezuju na isti izvor kreira se samo jedna instanca
- ❑ ***PropertyManager***
  - brine se za održavanje trenutne vrijednosti privezanog objekta
- ❑ ***BindingContext***
  - vodi evidenciju o svim objektima tipa *CurrencyManager* i *PropertyManager* na nekoj formi.
- ❑ ***Binding***
  - stvara i održava jednostavno povezivanje između pojedinačnog svojstva kontrole i svojstva nekog objekta (u nekoj listi objekata)
- ❑ ***BindingSource***
  - učahuruje izvor podataka i prati trenutnu poziciju zapisa. Ima funkcionalnosti *BindingContext* i *CurrencyManager*

# Razred *BindingSource*

## □ Glavni članovi

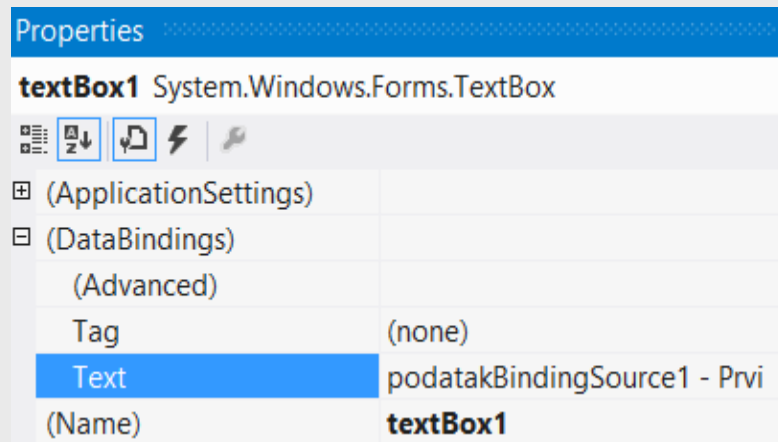
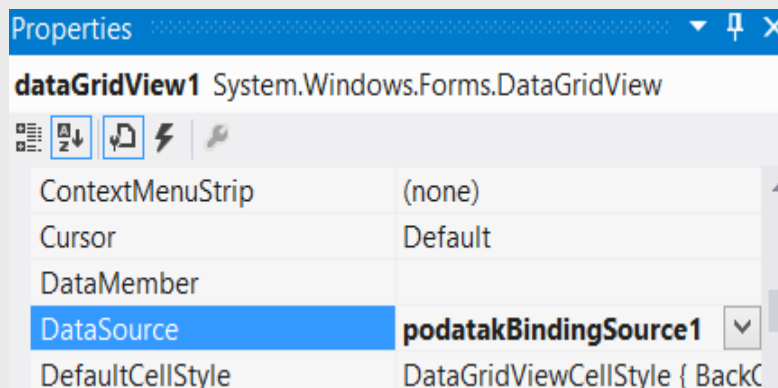
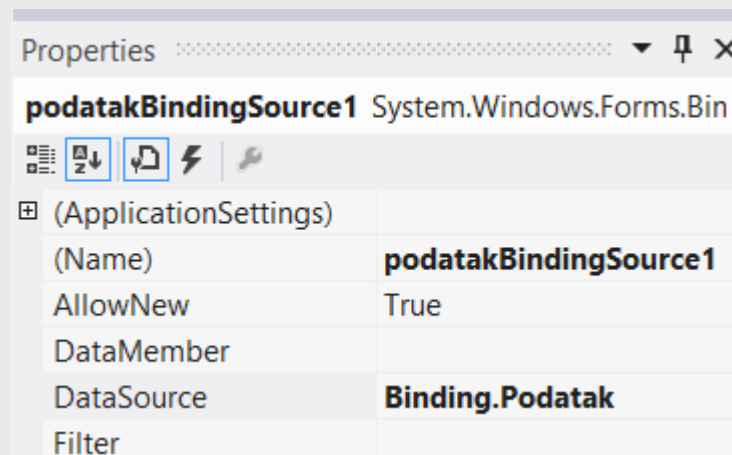
- *DataSource* – skup podataka
  - U dizajnu se postavlja na tip podatka
  - U pogonu na konkretne podatke
- *DataMember* – tablica skupa (ako skup ima više podskupova/tablica)

## □ Pojedina kontrola veže se

- na čitav BindingSource (složeno povezivanje) ili
- na jedan njegov član (jednostavno)

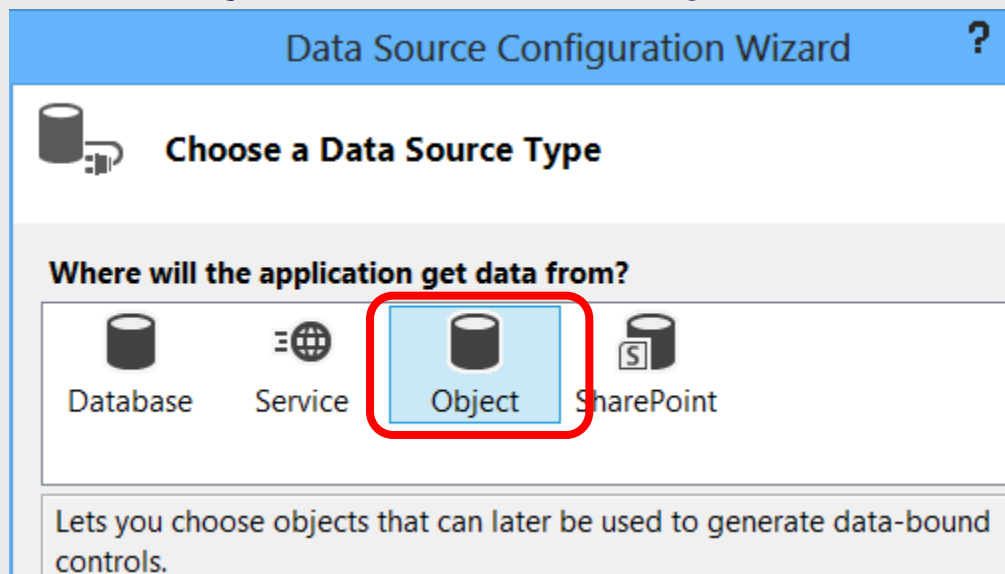
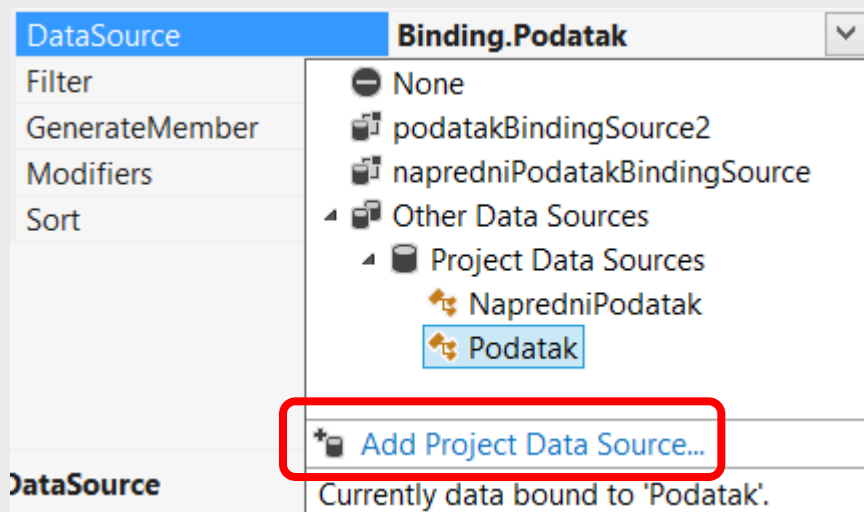
## □ Primjer na slici: Binding

- svojstva *BindingSource*
- postavljanje izvora za *DataGridView*
- postavljanje izvora za *TextBox*

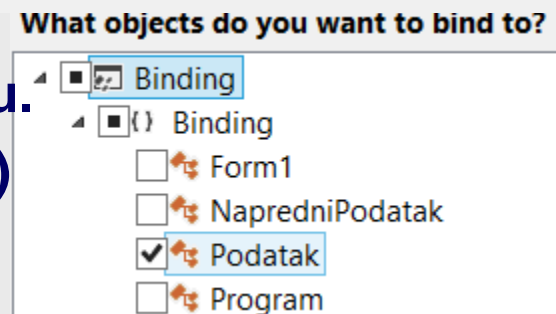


# Odabir tipa podatka za *BindingSource*

- ❑ Bira se jedan od tipova podataka unutar liste izvora u projektu
- ❑ Kada tip podatka nije u listi, opcijom *Add Project Data Source*, dodaje se tip u listu kao vrsta izvora *Object* i označava željeni tip



- ❑ Tipovi iz EF modela odabiru se po istom principu.
- ❑ U kôdu (najčešće u obradi događaja *Load* forme) se dohvate konkretni podaci te rezultat dohvata pridruži svojstvu *DataSource* izvora podataka



# Važniji članovi razreda *BindingSource*

## ❑ Svojstva

- `AllowEdit, AllowNew, AllowRemove` – indikatori da je postupak moguć
- `DataSource` – skup podataka
- `DataMember` – tablica skupa koja se povezuje
- `Count` – broj elemenata u listi podataka
- `object Current` - objekt aktualni element izvora
- `int Position` - indeks aktualnog elementa

## ❑ Događaji

- `CurrentChanged` – promjena `Current`
- `ItemChanged` – ažuriran aktualni element `List`
- `PositionChanged` – promjena `Position`

## ❑ Postupci

- `AddNew` – dodavanje novog elementa na izvor
- `CancelEdit` – opoziv uređivanja koje je u tijeku
- `EndEdit` - dovršetak uređivanja koje je u tijeku, pohrana na izvoru
- `Remove, RemoveAt(int index)` – brisanje elementa s izvora
- `MoveFirst, MoveLast, MoveNext, MovePrevious` – navigacija
- `ResetBindings, ResetCurrentItem` – ručno osvježavanje podataka

# Osvježavanje povezivanja

- ❑ Promjena kroz *BindingSource* ili kroz povezanu kontrolu automatski mijenja i izvor na koji je *BindingSource* povezan
- ❑ Promjena podataka u kôdu ne mora ažurirati povezane kontrole!

## ❑ Primjer: Binding

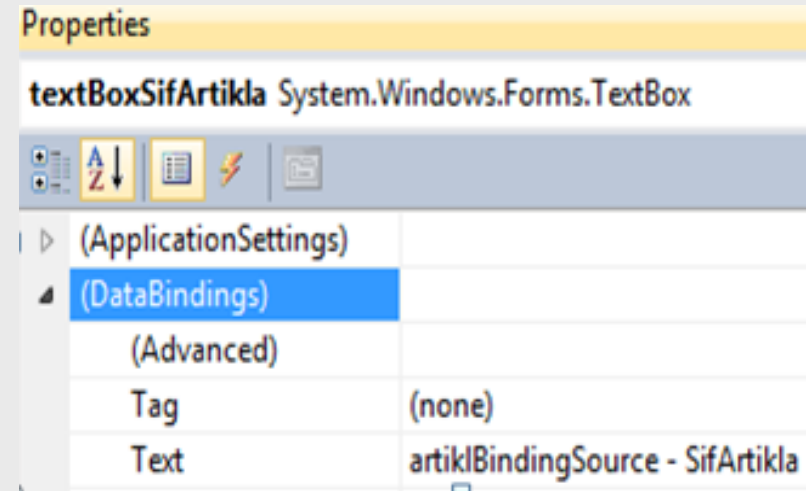
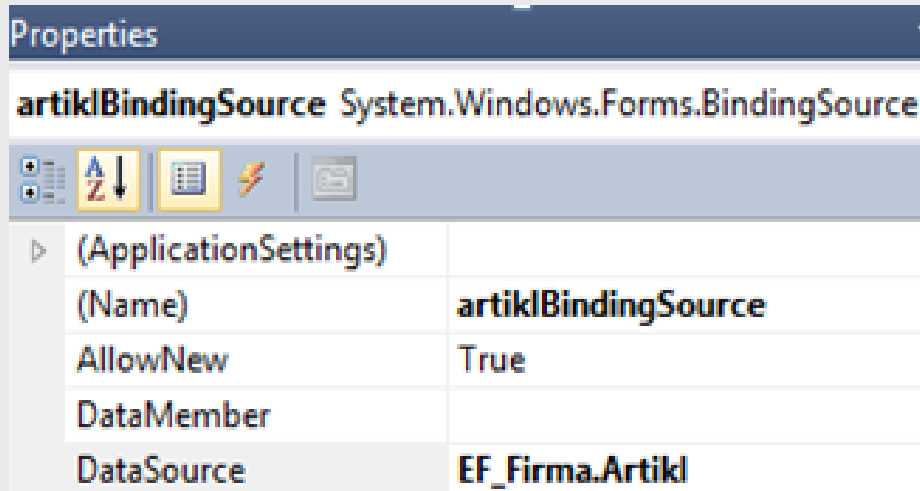
- *List<Podatak>* - dodavanje novih elemenata u listu ili ažuriranje elementa ne propagira promjene na formu
- *BindingList<Podatak>* - dodavanje elemenata u listu ili promjena reference za neku poziciju u listi propagira promjene na formu (događaj *ListChanged* u *IBindingList*)
- *BindingList<NapredniPodatak>* - isto kao i *BindingList<Podatak>* + propagira promjenu vrijednosti svojstva
  - *NapredniPodatak* : *INotifyPropertyChanged*
  - događaj *PropertyChanged* prilikom promjene svojstva objekta

# Povezivanje na podatke korištenjem EF modela

- ❑ ***BindingSource* se veže na neki tip entiteta iz EF modela**
  - Po uzoru na slajd „Odabir tipa podatka za *BindingSource*” (na svojstvu *DataSource* izvora podataka -> Add Project Data Source -> Object -> pa npr. *EF\_Firma.Artikl*)
- ❑ **Izvoru se pridružuju rezultati upita na bazu podataka**
  - Direktno povezivanje na *DbSet<T>* izazvalo bi rušenje programa
- ❑ ***DbSet<T>* se može kopirati u *List<T>* postupkom *ToList()***
  - koristi se kada se podaci samo čitaju (npr. lista za odabir stranog ključa)
  - Npr. `ctx.Partner.AsNoTracking().ToList()`
    - *AsNoTracking* da bi bilo brže
- ❑ **Kad se podaci mijenjaju koristi se svojstvo *.Local***
  - tipa *ObservableCollection<T>* predstavlja sve prethodno učitane elemente u kontekst i omogućava dodavanje novih i ažuriranje postojećih
  - potrebno je dodati postupak *ToBindingList()* (proširenje iz *System.Data.Entity*)
  - za razliku, WPF forme se vežu direktno na *Local*



# Primjer povezivanja na podatke iz BP korištenjem EF



## ❑ Primjer ADO\EF\_Firma\ArtikliForm

```
using System.Data.Entity; // zbog ToBindingList()
...
context = new FirmaEntities();
// pripremimo upit
var query = context.Artikl;
await query.LoadAsync(); //dovučemo podatke u kontekst
// prilagodimo podatke
BindingList<Artikl> artikli = context.Artikl.Local.ToBindingList();
// povežemo podatke
artiklBindingSource.DataSource = artikli;
```

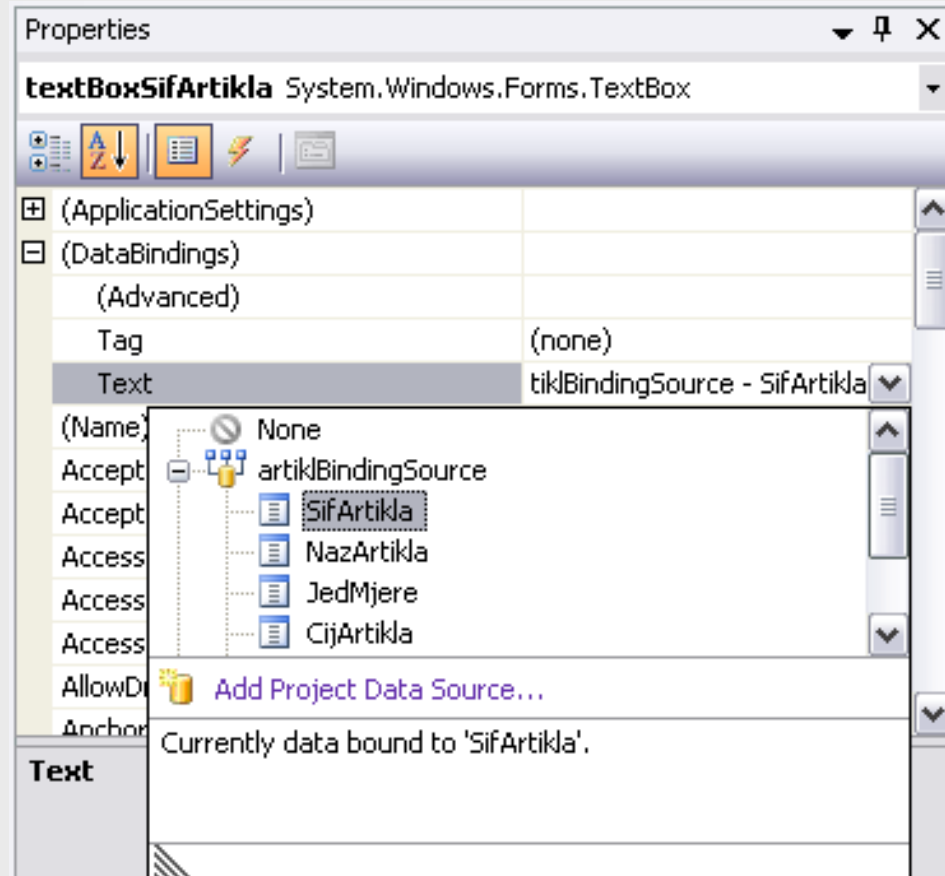
# Jednostavno povezivanje (Simple Binding)

## ❑ Povezivanje kontrole koja prikazuje jednu vrijednost

- svojstvo kontrole, npr. "Text"
- izvor podataka, npr. "artiklBindingSource"
- Svojstvo iz povezanog objekta , npr. "SifArtikla"

## ❑ Primjer: ADO\EF\_Firma\Artikl

- Properties – (Bindings):  
Text ili (Advanced)



# Jednostavno povezivanje dinamički

## ❑ Primjer: ADO\EF\_Firma\Artikl

- povezati neka svojstva u dizajnu a neka dinamički

```
textBoxJedMjere.DataBindings.Add("Text",  
    artiklBindingSource, "JedMjere", true);  
textBoxCijArtikla.DataBindings.Add("Text",  
    artiklBindingSource, "CijArtikla", true);  
checkBoxUsluga.DataBindings.Add("Checked",  
    artiklBindingSource, "ZastUsluga", true);  
pictureBoxArtikl.DataBindings.Add("Image",  
    artiklBindingSource, "SlikaArtikla", true,  
    System.Windows.Forms.DataSourceUpdateMode.Never);
```

- control.DataBindings – poveznice kontrole
- Binding – pojedinačna poveznica svojstvo – izvor - član/podatak
  - .DataBindings[ property ]
  - public Binding(string propertyName, Object dataSource, string dataMember);

# Složeno povezivanje (Complex Binding)

## ❑ Povezivanje složene kontrole s više redaka, postavljanjem

- DataSource – izvor podataka, npr. artiklBindingSource ili
- DataMember – u slučaju da izvor podataka ima više tablica
- DisplayMember – atribut, npr. NazivArtikla
  - kontrole ListBox i ComboBox – jedna vrijednost za redak

## ❑ Primjer: ADO\EF\_Firma\ArtikliForm

```
//u dizajnu...  
listBoxArtikli.DataSource = artiklBindingSource;  
listBoxArtikli.DisplayMember = "NazArtikla";  
//u kodu...  
artiklBindingSource.DataSource =  
    context.Artikl.Local.ToBindingList();
```

# Navigacija podacima

□ Primjer:

ADO\EF\_Firma\  
ArtiklForm

Pregled artikala

Knjiga "10 min. vodič kroz Outlook 2002"  
Knjiga "10 min. vodič kroz Front Page 2002"  
Knjiga "Ilustrator 10 za 24 sata"  
Knjiga "Access 2002 u 21 dan"  
Knjiga "Visual Basic.NET"  
Knjiga "C #" ekspert  
Knjiga "Windows Me za sveznalice" 19.4.2010 9:31:31  
Knjiga "Acrobat 5 za 24 sata"  
Knjiga "Office Xp za 24 sata"  
Knjiga "Photoshop 7 za 24 sata"  
Knjiga "Visual Basic.NET ab ovo"  
Knjiga "CorelDRAW 9 biblija"

Šifra: 5396

Naziv: Knjiga "C #" ekspert

Cijena: 219,0000

Mjera: kom

Usluga ☐

C# Biblioteka EKSPERT

<| < 9 od 1529 Unchanged > >| Spremi promjene

Dodaj novog Obriši označenog Otkazi promjene na trenutnom Ponovo učitaj

# Navigacija podacima vlastitim metodama

## ❏ ADO\EF\_Firma\ArtiklForm

```
private void buttonNext_Click(object sender, EventArgs e)
{
    artiklBindingSource.MoveNext();
}
```

## ❏ Pomak u izvoru automatski mijenja označenog u listi (jer je lista povezana s *artikliBindingSource*) – vrijedi i obrnuto

```
private void listBoxArtikli_SelectedIndexChanged(...) {
    UpdateDisplay();
}
```

```
private void UpdateDisplay(){//ažuriranje statusa na ekranu
    if (listBoxArtikli.SelectedIndex != -1 &&
        artiklBindingSource != null){

        labelPosition.Text =
            ((artiklBindingSource.Position + 1).ToString() +
            " od " + artiklBindingSource.Count.ToString());
        Artikl artikl = (Artikl)artiklBindingSource.Current;
        labelRowState.Text =
            context.Entry<Artikl>(artikl).State.ToString();
    }
}
```

# Dodavanje retka na kraj skupa podataka

## ❑ Primjer: ADO\EF\_Firma\ArtikliForm


```
private void buttonAdd_Click(object sender, EventArgs e)
{
    //završimo ažuriranje postojećeg elementa
    artiklBindingSource.EndEdit();

    //stvorimo novi artikl i dodamo ga u binding source
    Artikl artikl = new Artikl();
    artiklBindingSource.Add(artikl);

    //promijenimo koji element je trenutno označen
    artiklBindingSource.Position = artiklBindingSource.Count-1;
    textBoxSifArtikla.Focus();
}
```

## ❑ Redak dodan u memoriju, ali ne i u izvor podataka (pogledati sljedeći slajd)

# Izmjena retka u skupu podataka

- ❑ **Primjer:**  `ADO\EF_Firma\ArtiklForm.cs`
- ❑ **Uređivanje zapisa se provodi promjenom sadržaja vezanih kontrola**
  - podaci se automatski spremaju u memoriju klikom na drugu kontrolu, prelaskom na neki drugi zapis ili pozivom *EndEdit()* na izvoru podataka
- ❑ **Brisanje trenutnog zapisa:**  
`artiklBindingSource.RemoveCurrent();`
- ❑ **Podaci nisu izmijenjeni na izvoru!**
  - kontekst preko kojeg su podaci dohvaćeni vodi evidenciju o promjenama
  - Nakon promjene status retka u skupu podataka je `Modified`  
`Artikl artikl = (Artikl)artiklBindingSource.Current;`  
`Vrijednost context.Entry<Artikl>(artikl).State`  
`je Modified`
- ❑ **Za snimanje potrebno pozvati postupak *SaveChanges* (ili *SaveChangesAsync*) na kontekstu**
  - Snima sve promjene, ne samo pojedinačnu!
  - Snimanje se obavlja unutar transakcije



# Opoziv izmjena

- ❑ **Primjer:**  ADO\EF\_Firma\ArtiklForm.cs (buttonCancel\_Click)
- ❑ **Nove elemente treba odvojiti iz konteksta, a mijenjanim vratiti stanje u početno**
  - Obavlja se promjenom stanja entiteta
  - Prethodno potrebno dohvatiti entitet iz konteksta

```
Artikl artikl = (Artikl)artiklBindingSource.Current;  
DbEntityEntry<Artikl> entry = context.Entry<Artikl>(artikl);  
if (entry.State == EntityState.Added) {  
    entry.State = EntityState.Detached; // odvajanje  
}  
else {  
    entry.State = EntityState.Unchanged; // vrati stanje  
    artiklBindingSource.ResetCurrentItem();  
}
```

# Opoziv svih izmjena

- ❑ Dohvatiti sve elemente iz konteksta za koje se vodi evidencija o promjenama `S context.ChangeTracker.Entries()`
- ❑ Pronaći promijenjene, obrisane i nove (dodane)
- ❑ Svakom obrisanom ili mijenjanom elementu promijeniti stanje u `Unchanged`
  - Alternativno, za promijenjeni element može se pozvati postupak *Reload*
- ❑ Nove elemente odvojiti postavljanjem stanja na `Detached`

```
foreach (var entry in context.ChangeTracker.Entries()) {  
    if (entry.State == EntityState.Added) {  
        entry.State = EntityState.Detached;  
    }  
    else if (entry.State == EntityState.Deleted ||  
            entry.State == EntityState.Modified) {  
        entry.State = EntityState.Unchanged;  
    }  
}
```

- ❑ Alternativno: odbaciti kontekst i instancirati novi

# Validacija na formi

## ❑ Obrada događaja *Validating* na pojedinoj kontroli

```
private void textBoxNazArtikla_Validating(object sender,
CancelEventArgs e) {
    if (string.IsNullOrEmpty(textBoxNazArtikla.Text)) {
        errorProviderArtikl.SetError(textBoxNazArtikla,
            "Form: Naziv artikla ne smije biti prazan");
        e.Cancel = true;
    }
    else{
        errorProviderArtikl.SetError(textBoxNazArtikla, string.Empty);
        e.Cancel = false;
    }
}
```

## ❑ Za validaciju svih kontrola na formi pozvati *ValidateChildren* forme

# Validacija korištenjem sučelja *IDataErrorInfo* (1)

## ❑ Za konkretni entitet implementirati sučelje *IDataErrorInfo*

- *Error* – vraća opis pogreške za objekt (prazan string ako nema pogreške)
- *Item* – vraća opis pogreške za konkretno svojstvo objekta (prazan string ako nema pogreške)

## ❑ Primjer: ADO\EF\_Firma\ArtiklForm.cs i ADO\EF\_Firma\Partial\Artikl.cs

```
public partial class Artikl : IDataErrorInfo {  
    public string Error{  
        get { ... }  
    }  
  
    public string this[string columnName]{ // „Item”  
        get {  
            ...  
            if (columnName == "CijArtikla" && CijArtikla <= 0)  
            {  
                return "Cijena artikla mora biti veća od 0";  
            }  
            ...  
        }  
    }  
}
```

# Validacija korištenjem sučelja *IDataErrorInfo* (2)

- ❑ Na kontroli tipa *ErrorProvider* postaviti svojstvo *DataSource* na izvor podataka koji implementira *IDataErrorInfo*:

```
errorProviderArtikl.DataSource = artiklBindingSource;
```

- ❑ *ErrorProvider* poziva indekserski predaju naziv svojstva povezanog s određenom kontrolom
- ❑ Prije snimanja provjeriti postoji li pogreška na konkretnom entitetu

```
error = ((IDataErrorInfo)artiklBindingSource.Current).Error;
```

# Kontrola *DataGridView*

## ❑ Mrežna/tablična obrada podataka s različitih vrsta izvora

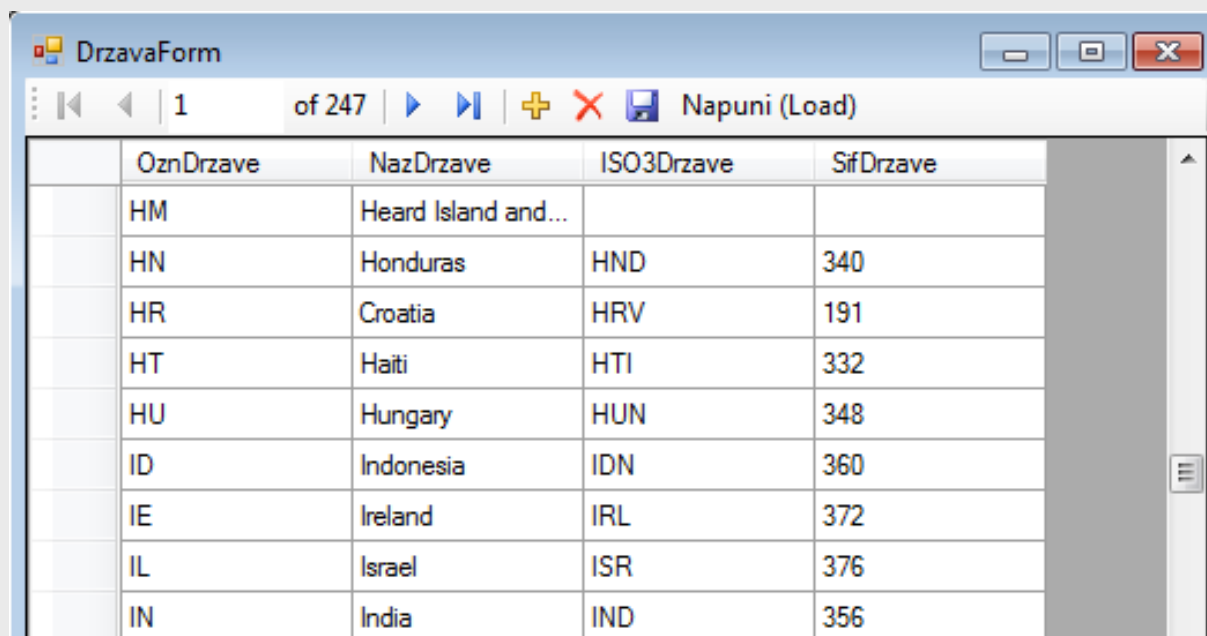
- Liste (*IList*), EF model, skupovi podataka (*DataSet*, *DataTable*), *bindingsource*

## ❑ Svojstva *DataSource* i *DataMember*

- *DataSource* – izvor podataka
- *DataMember* – naziv liste ili tablice izvora (kada izvor ima više tablica ili lista)

## ❑ Primjer: ADO\EF\_Firma – DrzavaForm

- `DataSource = drzavaBindingSource, DataMember = ""`

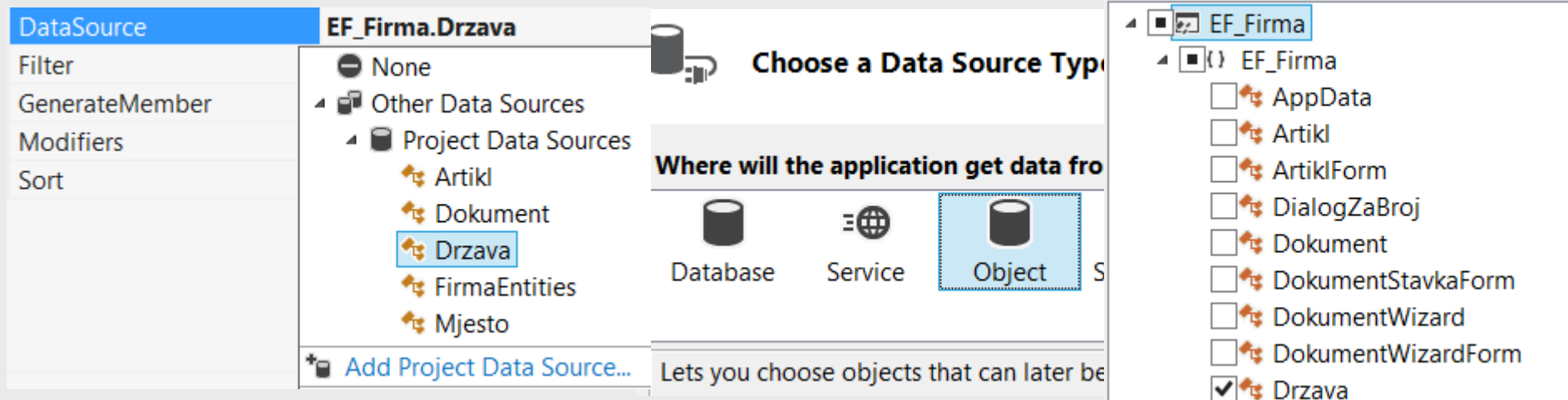


	OznDrzave	NazDrzave	ISO3Drzave	SifDrzave
	HM	Heard Island and...		
	HN	Honduras	HND	340
	HR	Croatia	HRV	191
	HT	Haiti	HTI	332
	HU	Hungary	HUN	348
	ID	Indonesia	IDN	360
	IE	Ireland	IRL	372
	IL	Israel	ISR	376
	IN	India	IND	356

# Primjer povezivanja podataka na *DataGridView* (1)

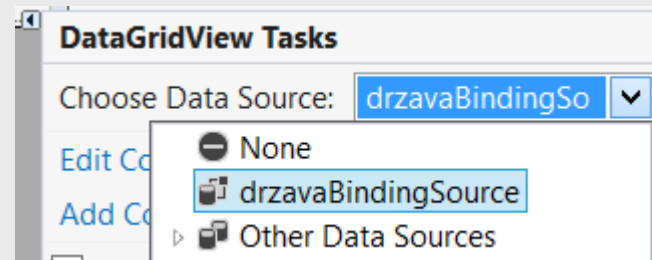
## ❑ Primjer: ADO\EF\_Firma - DrzavaForm

- Na formu dodati *DataGridView* i izvor podataka (*BindingSource*)
- Za svojstvo *DataSource* izvora podataka postaviti *EF\_Firma.Drzava*
- Ako nije u listi izvora unutar projekta odabrati *Add Project Data Source*, za vrstu odabrati *Object*, a zatim označiti razred *Drzava*



## ❑ Kontroli *DataGridView* postaviti *DataSource* na konkretni *BindingSource*

- ❑ **Napomena:** Postupak se može skratiti tako da se kod odabira izvora za *DataGridView* odmah ide na *Add Project Data Source* – automatski doda novi *BindingSource*



# Primjer povezivanja podataka na *DataGridView* (2)

## ❑ Primjer: ADO\EF\_Firma – DrzavaForm

- Dohvat podataka prilikom učitavanja forme
- Prethodno definiranom izvoru podataka (drzavaBindingSource) pridružuju se konkretni podaci iz baze podataka
  - Ako je moguće poželjno koristiti `async` i `await` da ne blokiramo glavnu nit

```
private async void DrzavaForm_Load(...) {
    await LoadData();
}
private async Task LoadData() {
    ...
    context = new FirmaEntities();
    var query = context.Drzava.
        OrderBy(d => d.OznDrzave); //definiramo upit
    await query.LoadAsync(); //napunimo podatke u kontekst
    // uzmemo podatke koji su dosad učitani
    drzavaBindingSource.DataSource =
        context.Drzava.Local.ToBindingList();
}
```



# Članovi DataGridView

## ❑ Svojstva

- Columns – kolekcija stupaca (*DataGridViewColumnCollection*)
  - SortMode svojstvo stupca – postavljanje načina sortiranja prilikom klika na zaglavlje { Automatic, Programmatic, NotSortable}
- Rows – kolekcija redaka (*DataGridViewRowCollection*)
- SelectedCells – kolekcija označenih ćelija (*DataGridViewSelectedCellCollection*)
- ColumnHeadersVisible, RowHeadersVisible – vidljivost zaglavlja
- CurrentCell, CurrentRow – trenutno aktivna ćelija/redak
- DefaultCellStyle – definira izgled ćelije
- ReadOnly – oznaka da je dozvoljeno uređivanje

## ❑ Indeksir

- [columnIndex, rowIndex] – dohvat ćelije u retku indeksa rowIndex i stupcu indeksa columnIndex

## ❑ Postupci

- BeginEdit, EndEdit – započinjanje / završetak promjena

# Događaji i stupci DataGridView

## ❑ Događaji

- `CurrentCellChanged` – promjena fokusa aktivne ćelie
- `CellBeginEdit`, `CellEndEdit` – okida se pri započinjanju/završavanju promjena u ćeliji
- `CellValueChanged` – okida se nakon završetka promjena unutar ćelije
  - argument `DataGridViewCellEventArgs` – podaci (svojstva `ColumnIndex`, `RowIndex`)

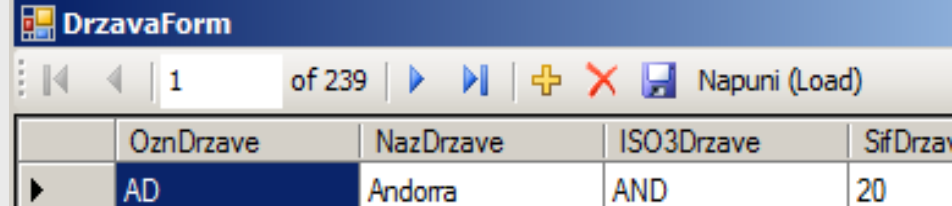
## ❑ Stupac *DataGridView* kontrole (iz *DataGridViewColumn*) može biti

- Tekstovni element (*DataGridViewTextBoxColumn*)
- Padajući izbornik (*DataGridViewComboBoxColumn*)
- Slika (*DataGridViewImageColumn*)
- ...

## ❑ Neka svojstva elemenata *DataGridViewColumn*

- `HeaderText`, `ReadOnly`, `Visible`, `DefaultCellStyle`, ...
- Svojstva elementa *DataGridViewComboBoxColumn*
  - `DataSource`, `DisplayMember`, `ValueMember`, ...

# BindingNavigator



OznDrzave	NazDrzave	ISO3Drzave	SifDrzave
AD	Andorra	AND	20

## ❑ BindingNavigator – komponenta grafičkog sučelja za navigaciju i rukovanje povezanim podacima

### ■ Svojstva:

- BindingSource – povezljivi izvor s podacima
- MoveFirstItem, MoveLastItem, MoveNextItem, MovePreviousItem – ToolStripItem funkcionalnosti navigacije
- PositionItem - ToolStripItem za prikaz pozicije aktivnog zapisa
- AddNewItem, DeleteItem - ToolStripItem funkcionalnosti dodavanja i brisanja

### ■ Događaji: (zanimljiviji su oni elemenata kontrole ili vlastitih gumba)

## ❑ Primjer: ADO\EF\_Firma\DrzavaForm

```
private async void drzavaBindingNavigatorSaveItem_Click(...)
{
    drzavaBindingSource.EndEdit();
    //provjeriti da li su podaci valjani i ako da
    await context.SaveChangesAsync();
}
```

# Sortiranje i filtriranje

---

# Sortiranje i filtriranje podataka u EF-u (1)

## ❑ Primjer: ADO\EF\_Firma\SortFilterForm

Sort & Filter

Filter

Naziv mjesta ▾

brod

Filters:

Naziv mjesta : brod

Sort

Naziv države ▾

Uzlazno ▾

Naziv mjesta ▾

Silazno ▾

Pbr mjesta ▾

Uzlazno ▾

▾

Uzlazno ▾

Apply sort and filter

	PBR	Naziv	Poštanski naziv mjesta	Ozn države	Naziv države
▶	77265	Martin Brod		BA	Bosnia and Herzegovina
	76313	Brodac		BA	Bosnia and Herzegovina
	73309	Brod (kod Foče)		BA	Bosnia and Herzegovina
	74450	Bosanski Brod		BA	Bosnia and Herzegovina
	51301	Zapolje Brodsko	Brod na Kupi	HR	Croatia
	51301	Zamost Brodski	Brod na Kupi	HR	Croatia

# Sortiranje i filtriranje podataka u EF-u (2)

## ❏ Primjer: ADO\EF\_Firma\SortFilterForm

- upit za dohvat podataka o mjestima bez praćenja promjena
- *Include("Drzava")* – odmah se dohvaćaju vezani podaci iz *Drzava* za mjesto
  - inače bi se za svaki redak u mreži izvodio novi upit
- upit se proširi dijelom za sortiranje i filtriranje te se rezultat pretvori u listu i poveže na formu
  - Koristimo *await* + asinkronu verziju *ToListAsync*

```
private async void buttonSortAndFilter_Click (...){  
    var query = context.Mjesto.  
        Include(m => m.Drzava) .AsNoTracking();  
    query = ApplyFilter(query);  
    IOrderedQueryable<Mjesto> sortedQuery = ApplySort(query);  
    if (sortedQuery != null)  
        mjestoBindingSource.DataSource = await sortedQuery.ToListAsync();  
    else  
        mjestoBindingSource.DataSource = await query.ToListAsync();  
}
```

# Sortiranje podataka

## ❏ Primjer: ADO\EF\_Firma\SortFilterForm

- Građenje upita kombiniranjem postupaka *OrderBy*, *OrderByDescending*, *ThenBy*, *ThenByDescending*
- Rezultat postupka je *IOrderedQueryable<T>*
- Upit se izvršava tek kod dohvata prvog podatka

```
private IOrderedQueryable<Mjesto> ApplySort
                                   (IQueryable<Mjesto> query) {
    IOrderedQueryable<Mjesto> sortedQuery = null;
    ...
    sortedQuery = query.OrderByDescending (m => m.PostBrMjesta);
    ...
    sortedQuery = sortedQuery.ThenBy (m => m.NazMjesta);
    ...
    return sortedQuery ;
}
```

# Filtriranje podataka

## ❑ Primjer: ADO\EF\_Firma\SortFilterForm

- Građenje upita kombiniranjem postupka *Where*
- Argument i rezultat postupka su tipa `IQueryable<T>`
- Upit se izvršava tek kod dohvata prvog podatka

```
private IQueryable<Mjesto> ApplyFilter(IQueryable<Mjesto>
query) {
    ...
    int pbr = int.Parse(vrijednost);
    query = query.Where(m => m.PostBrMjesta == pbr);
    ...
    query = query.Where(m =>
        m.Drzava.NazDrzave.Contains(vrijednost));
    ...
    return query;
}
```



# Složene zaslonske maske

---

# Što je master-detail?

- ❑ **Zaglavlje (Master)** – dio forme na kojem se ažurira pojedinačni zapis
- ❑ **Detalji (Details)** – dio forme s podacima zavisnim o zaglavlju
- ❑ **Veza 1:N – 1 podatak u zaglavlju ima N podataka u detaljima**
  - Primjer: (master) *Dokument* 1:N (detail) *Stavka*
    - U edmx modelu veza nazvana *Stavke* umjesto inicijalno *Stavka*
- ❑ **Ostvarenje: Izvor podataka za detalje je dio izvora podataka za zaglavlje**
- ❑ **Potrebna preinaka u već generiranom EF modelu**
  - HashSet nije pogodan za povezivanje detalja
  - U konstruktoru razreda *Dokument* (*Firma.edmx* -> *Firma.tt* -> *Dokument.cs*) promijeniti

```
this.Stavke = new HashSet<Stavka>(); u
this.Stavke = new BindingList<Stavka>();
```
  - `public virtual ICollection<Stavka> Stavke { get; set; }` promijeniti u

```
public virtual BindingList<Stavka> Stavke { get; set; }
```
- ❑ **Ponovnim generiranjem modela promjene će se poništiti.**
  - Alternativa ponovnoj promjeni je modificiranje T4 (.tt) predloška

# Zaslonska maska oblika zaglavlje-stavke

❑ Primjer: ADO\EF\_Firma\DokumentStavkaForm

- Prikaz i obrada detalja sinkronizirani s promjenama zaglavlja

DokumentStavka

482 of 857

Id dokumenta: 2115 Vrsta: 0 Broj: 1197 Datum: 6. 2.2010.

Partner: ETIKET (5017600) Porez: 0,22 [0 -1]

Prethodni: X Iznos: 128.614,90 kn

	Naziv artikla	Količina	Jedinična cijena	Rabat	Ukupna cijena
▶	Stalak za CD NJAMA, Box 75 (75 ...	3,00	114,00 kn	3,00%	331,74 kn
	Knjiga "Office 2003 za Windows"	1,00	165,00 kn	4,00%	158,40 kn
	MBO XBIT, s. 939, AN8, nForce 4,...	4,00	1.187,00 kn	2,00%	4.653,04 kn
	Torba za notebook, TARGUS TS...	5,00	139,00 kn	7,00%	646,35 kn
	Mobilni uređaj NUKIA 6111, GPR...	2,00	1.429,00 kn	5,00%	2.715,10 kn
	Torba za notebook, PORT, Chica...	1,00	199,00 kn	9,00%	181,09 kn
	Mobilni uređaj NUKIA 7390, GPR...	3,00	2.849,00 kn	10,00%	7.692,30 kn
	PRODULJENJE GARANCIJE ZA ...	5,00	542,00 kn	2,00%	2.655,80 kn
	MP3/FM/Video player, LGG MF-F...	1,00	899,00 kn	2,00%	881,02 kn

# Izračunata polja

## ❑ Primjer: ADO\EF\_Firma\Partial\Dokument.cs

- Entitet *Dokument* proširen svojstvom za lakši prikaz u padajućoj listi

```
partial class Dokument{  
    public string LookupText {  
        get { return string.Format("{0} - {1} - {2}",  
            DatDokumenta.ToString("dd.MM.yyyy"), IdDokumenta,  
            IznosDokumenta);  
        ...  
    }  
}
```

## ❑ Primjer: ADO\EF\_Firma\Partial\Stavka.cs

- Entitet *Stavka* proširen novim svojstvom

```
partial class Stavka{  
    public decimal UkupnaCijena{  
        get{  
            return this.KolArtikla * this.JedCijArtikla *  
                (1 - this.PostoRabat);  
        }  
        ...  
    }  
}
```

## ❑ Slično napravljeno i za ostale entitete (npr. Partner)

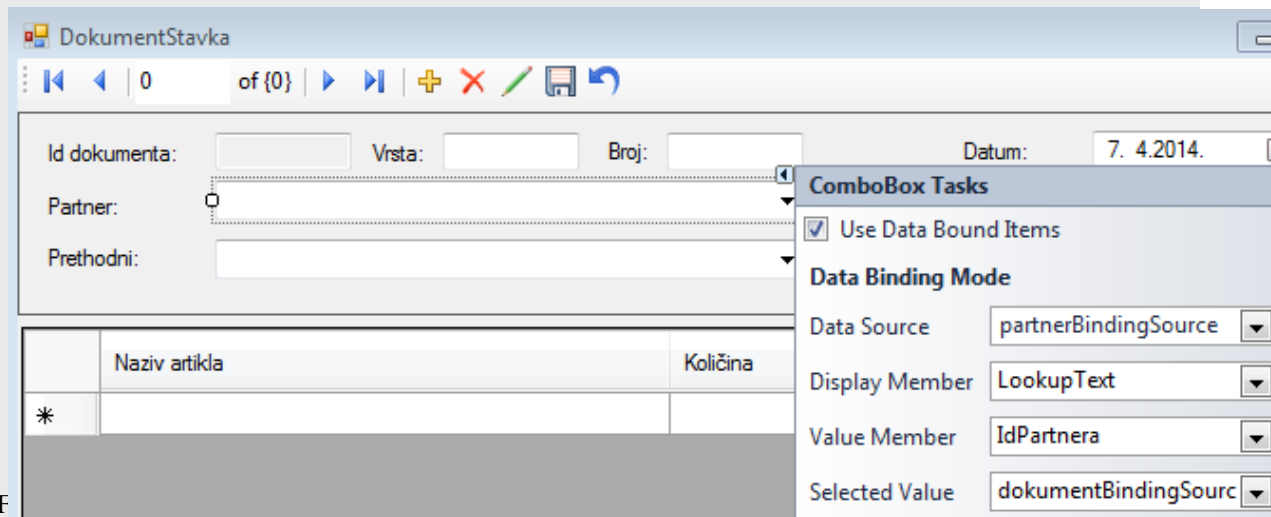
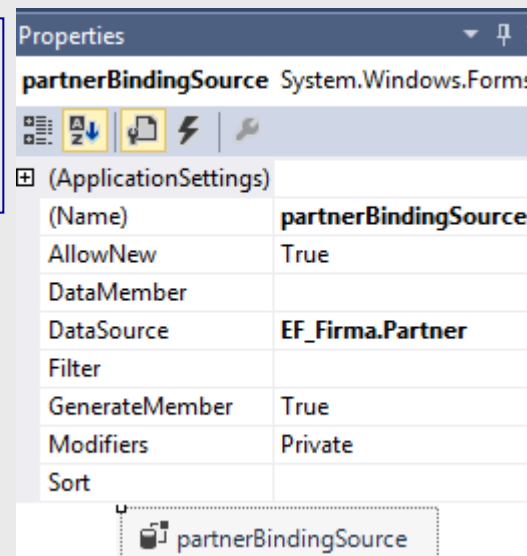
# Odabir vrijednosti stranog ključa

## ❑ Primjer ADO\EF\_Firma\DokumentStavkaForm

- Potrebno dodati izvor podataka koji će se naknadno popuniti vrijednostima iz baze podataka (ili nekog drugog izvora)

```
partnerBindingSource.DataSource = await  
    context.Partner  
        .AsNoTracking().ToListAsync();
```

- Odabire se svojstvo za prikaz (*DisplayMember*), za vrijednost (*ValueMember*) te svojstvo koje predstavlja strani ključ (*SelectedValue*)
  - Display Member: izvedeno polje LookupText
  - Value Member: IdPartnera
  - Selected Value: dokumentBindingSource.IdPartnera



# Odabir vrijednosti stranog ključa u rekurzivnoj vezi

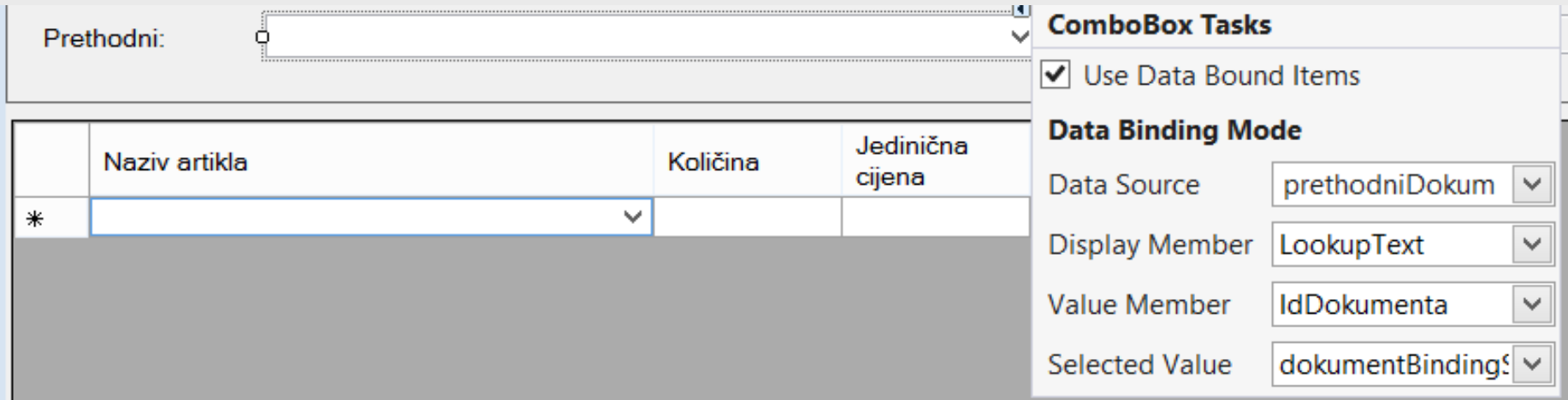
## ❑ Primjer ADO\EF\_Firma\DokumentStavkaForm


## ❑ Dodavanjem novog izvora podataka


- prethodniDokumentBindingSource sa svojstvom DataSource = EF\_Firma.Dokument
- ne može se koristiti isti kao za osnovne podatke, jer bi ograničio izbor samo na trenutno dohvaćene i povezane podatke

## ❑ Odabire se svojstvo za prikaz (*DisplayMember*) i za vrijednost (*ValueMember*) te svojstvo koje predstavlja strani ključ (*SelectedValue*)

- Display Member: izvedeno polje LookupText
- Value Member: IdDokumenta
- Selected Value: dokumentBindingSource.IdPrethDokumenta







Prethodni: 

	Naziv artikla	Količina	Jedinična cijena
*			

**ComboBox Tasks**

- ☒ Use Data Bound Items

**Data Binding Mode**

Data Source	prethodniDokum 
Display Member	LookupText 
Value Member	IdDokumenta 
Selected Value	dokumentBindingSource.IdPrethDokumenta 

# Sinkronizacija stavki

- ❑ Izvor za zaglavlje: *dokumentBindingSource*
- ❑ Izvor za detalje: *stavkeBindingSource*
  - DataSource = dokumentBindingSource
- ❑ Mreža stavki *dataGridViewStavke* vezana na *stavkeBindingSource*
- ❑ Primjer:  ADO\DokumentStavka – dizajn

```
stavkeBindingSource.DataMember =  
"Stavke";  
stavkeBindingSource.DataSource =  
this.dokumentBindingSource;  
...  
dataGridViewStavke.DataSource =  
this.stavkeBindingSource;
```

Properties

**dokumentBindingSource** System.Windows.Forms.BindingSource

(ApplicationSettings)	
(Name)	dokumentBindingSource
AllowNew	True
DataMember	
DataSource	EF_Firma.Dokument
Filter	
GenerateMember	True
Modifiers	Private
Sort	

Properties

**stavkeBindingSource** System.Windows.Forms.BindingSource

(ApplicationSettings)	
(Name)	stavkeBindingSource
AllowNew	True
DataMember	Stavke
DataSource	dokumentBindingSource
Filter	
GenerateMember	True
Modifiers	Private
Sort	

# Referentne vrijednosti stavke

## ❑ *dataGridViewStavke – Tasks – Edit Columns*

- postavljamo *ColumnType* na *DataGridViewComboBoxColumn* i *DisplayStyle* na *ComboBox* (alternativno – *Nothing* pa se padajuća lista pojavljuje samo pri ažuriranju)
- Postavljamo *DataPropertyName*, *DataSource*, *DisplayMember* i *ValueMember*
- za *DataSource* odabran poseban *BindingSource* nad entitetom *Artikl*

The screenshot displays two windows from the Visual Studio IDE. The 'Edit Columns' dialog is in the foreground, showing a list of columns for a DataGridView. The 'Selected Columns' list includes 'Naziv artikla', 'Količina', 'Jedinična cijena', and 'Rabat'. The 'Bound Column Properties' section is expanded for the 'Naziv artikla' column, showing the following properties:

Bound Column Properties	
SortMode	Automatic
<b>Data</b>	
DataPropertyName	SifArtikla
DataSource	artiklBindingSource
DisplayMember	NazArtikla
Items	(Collection)
ValueMember	SifArtikla
<b>Design</b>	
(Name)	sifArtiklaDataGridViewTextBo
ColumnType	DataGridViewComboBoxCol


Below the properties table, the '(Name)' property is highlighted with a description: 'Indicates the name used in code to identify the object.' The 'OK' and 'Cancel' buttons are at the bottom right of the dialog.

In the background, the 'Properties' window for 'artiklBindingSource' is visible, showing the following properties:

Properties	
artiklBindingSource System.Windows.Forms.Bindin	
(ApplicationSettings)	
(Name)	artiklBindingSource
AllowNew	True
DataMember	
DataSource	EF_Firma.Artikl
Filter	
GenerateMember	True
Modifiers	Private
Sort	



# Početne vrijednosti stavki

- ❑ Specifičnost funkcioniranja *DataGridView* kod dodavanja stavke uzrokuje stvaranje stavke s praznim (null), odnosno pretpostavljenim vrijednostima
- ❑ Problem ako referentna vrijednost za padajuću listu ne može biti null, odnosno ako nema elementa s pretpostavljenom vrijednošću.
  - Uzrokuje pogrešku oblika “*datagridviewcomboboxcell value is not valid*”
  - Rješenje – dodati element s pretpostavljenom vrijednosti za određeni tip
    - Konkretno artikl s vrijednosti 0 ( može i *default(int)*)
  - Primjer  ADO\EF\_Firma\DokumentStavkaForm.cs

```
private async Task LoadDataSources() {  
    context = new FirmaEntities();  
    List<Artikl> artikli = await context.Artikl  
                                .AsNoTracking().ToListAsync();  
    artikli.Insert(0, new Artikl {  
        SifArtikla = 0,  
        NazArtikla = "-----Odaberite artikl-----"  
    });  
    artiklBindingSource.DataSource = artikli;  
    ...  
}
```

# Ažuriranje jedinične cijene artikla

## ❑ Primjer: ADO\EF\_Firma\DokumentStavkaForm

- Ažuriranje jedinične cijene artikla i iznosa dokumenta pri promjeni artikla

```
private async void dataGridViewStavke_CellEndEdit(...) {
    stavkeBindingSource.EndEdit();
    Stavka stavka = (Stavka)stavkeBindingSource.Current;
    if (e.ColumnIndex == 0) { // stupac s artiklom
        //dohvati cijenu artikla
        Artikl artikl = await context.Artikl
                                .FindAsync(stavka.SifArtikla);
        stavka.JedCijArtikla = artikl.CijArtikla;
    }
    AzuriraCijenuDokumenta();
}

private void AzurirajCijenuDokumenta() {
    Dokument d = (Dokument)dokumentBindingSource.Current;
    d.IznosDokumenta = d.Stavke.Sum(s => s.UkupnaCijena) *
                        (1 + d.PostoPorez);
}
```

# Dodavanje novog dokumenta

## ☐ Primjer: ADO\EF\_Firma\DokumentStavkaForm

- Automatski korištenjem navigatora – dodaje se novi podatak u izvor podataka



## ☐ Dodatno postavimo predviđeni (default) broj dokumenta i ažuriramo prikaz navigatora

```
private void bindingNavigatorAddNewItem_Click(...) {  
    ((Dokument)dokumentBindingSource.Current).BrDokumenta =  
        context.Dokument.  
            Max(d => d.BrDokumenta) + 1;  
    UpdateDisplay(true);  
}
```

## ☐ Napomena: Podaci još nisu pohranjeni u bazi!

# Spremanje podataka

## ❑ Primjer: ADO\EF\_Firma\DokumentStavkaForm

- Prihvatiti promjene za kontrole koje su trenutno aktivne
- Izračunati cijenu dokumenta
- Provjeriti ima li pogrešaka na dokumentu (koristi se sučelje IDataErrorInfo)
- Spremiti promjene u kontekstu pozivom postupka *SaveChanges[Async]*

```
private async void toolStripButtonSave_Click(...) {  
    stavkeBindingSource.EndEdit(); dokumentBindingSource.EndEdit();  
    AzurirajCijenuDokumenta();  
    Dokument dokument = (Dokument)dokumentBindingSource.Current;  
    string error = dokument.Error; //Error iz IDataErrorInfo  
    if (!string.IsNullOrEmpty(error)) {  
        ...  
    }  
    else {  
        await context.SaveChangesAsync();  
    }  
    ...  
}
```

- Nakon snimanja EF automatski ažurira vrijednosti samopovećavajućih primarnih ključeva, kao i ključeve roditelja u stavkama.

# Brisanje zapisa (zaglavlja skupa s detaljima)

## ❑ Primjer: ADO\EF\_Firma\DokumentStavkaForm

- Označiti trenutni dokument za brisanje pomoću *context.Entry* ili ga izbaciti iz trenutnog izvora podataka
- Spremiti promjene u kontekstu pozivom postupka *SaveChanges*
- Stavke se brišu kaskadno (postavljeno u modelu)

## ❑ Na *BindingNavigatoru* svojstvo *DeleteItem* postavljeno na (*none*), a obrađuje se klik – u protivnom uvijek dolazi do brisanja

```
private async void bindingNavigatorDeleteItem_Click(...) {  
    DialogResult result = MessageBox.Show  
        ("Želite li obrisati zapis",  
        "Brisanje zapisa", MessageBoxButtons.YesNo);  
    if (result == System.Windows.Forms.DialogResult.Yes) {  
        dokumentBindingSource.RemoveCurrent();  
        await context.SaveChangesAsync();  
    }  
}
```

# Brisanje pojedinačne stavke

## ❑ Zaobilazno rješenje zbog načina rada EF-a i DataGridViewa

- Brisanjem retka umjesto brisanja stavke briše se vrijednost stranog ključa

## ❑ Rješenje: Obraditi događaj za brisanje označenih redaka

```
private void dataGridViewStavke_UserDeletingRow (...) {  
    List<Stavka> stavkeZaBrisanje = new List<Stavka>();  
  
    for (int i=0; i < dataGridViewStavke.SelectedRows.Count; i++){  
        DataGridViewRow row = dataGridViewStavke.SelectedRows[i];  
        stavkeZaBrisanje.Add((Stavka)row.DataBoundItem);  
    }  
    foreach(Stavka stavka in stavkeZaBrisanje){  
        context.Stavka.Remove (stavka);  
    }  
    e.Cancel = true;//otkaži normalno izvođenje ovog događaja  
    AzurirajCijenuDokumenta();  
}
```

# Opoziv izmjena

## ❑ Primjer: ADO\EF\_Firma\DokumentStavkaForm

- Dohvatiti sve entitete za koje se evidentira promjena
- Pronaći nove (dodane) i odspojiti ih
- promijeniti stanje označenih za brisanje i promijenjenih na *Unchanged*

```
private void CancelChanges() {  
    foreach (var entry in context.ChangeTracker.Entries()) {  
        if (entry.State == EntityState.Added)  
        {  
            entry.State = EntityState.Detached;  
        }  
        else if (entry.State == EntityState.Deleted  
                || entry.State == EntityState.Modified)  
        {  
            entry.State = EntityState.Unchanged;  
        }  
        ...  
    }  
}
```

- ❑ Nakon otkazivanja promjena, na izvoru pozvati `ResetCurrentItem()` jer entiteti ne implementiraju `INotifyPropertyChanged`



# Zadaci za vježbu

- ☐ Doraditi *Artikl* dodavanjem tablice *JedinicaMjere* i veze s artiklom
- ☐ Ugraditi sortiranje i filtriranje podataka u *Drzava*
- ☐ Ugraditi sortiranje i filtriranje podataka u *Artikl*
- ☐ Napisati program koji će ažurirati tablicu *Artikl* tako da obradi sadržaj mape u kojoj se nalaze slike naziva oblika <IdArtikla>.jpg
- ☐ U formi *DokumentStavke* u *DataGridView* za prikaz stavki dodati stupac za šifru artikla te je povezati na isti izvor kao i naziv artikla
- ☐ Implementirati validaciju za primjer *DokumentStavke*



# Reference

## ❑ Povezivanje podataka

- Databinding and Windows Forms

[http://msdn.microsoft.com/en-us/library/c8aebh9k\(v=VS.100\).aspx](http://msdn.microsoft.com/en-us/library/c8aebh9k(v=VS.100).aspx)

- Interfaces Related to DataBinding

<http://msdn.microsoft.com/en-us/library/41e17s4b.aspx>

- MS Patterns & Practices - Chapter 2 — Handling Data

<http://msdn.microsoft.com/en-us/library/ff647254.aspx>