

# **Web aplikacije ASP.NET Web Forms**

---

**2014/15.10**

# Izrada web aplikacija pomoću ASP.NET-a

## ❑ Što je web aplikacija?

- Programska aplikacija kojoj se pristupa preko internetskog preglednika ili drugog programa koji implementira HTTP (Hyper Text Transfer Protocol)

## ❑ Da li je skup web stranica ujedno i web aplikacija?

- Aplikacija koristi programsku logiku da bi prikazala sadržaj korisniku
- Kod web aplikacija obično se izvršava neki programski kod na serveru
- Primjer: statički cjenik nije aplikacija; dinamički cjenik je aplikacija

## ❑ Primjer web aplikacije: <http://ahyco.fer.hr>, s funkcijama

- Identifikacija (autorizacija) korisnika
- Pisanje provjere, pregled rezultata, ...
- Baza podataka + programska logika za generiranje web stranica

## ❑ Elementi web stranice

- HTML (HyperText Markup Language): osnovni jezik za definiranje web stranica
- JavaScript: jezik za klijentske skripte koje izvodi preglednik
- C# kôd (ili kôd u nekom drugom jeziku) koji se izvodi na poslužitelju

# Karakteristike web aplikacija

- Izvršava se i na poslužitelju i na klijentu
  - Klijentski dio mora biti napisan u nekom od jezika koji Web preglednik podržava (HTML, JavaScript).
- Ograničen pristup resursima na strani klijenta (npr. JavaScript koji se izvršava unutar preglednika ne može čitati s korisnikovog diska)

## ☐ Prednosti

- Korisnik može biti bilo tko s pristupom Internetu
  - nema instalacijske procedure na strani klijenta
  - koriste se na bilo kojem OS, koji ima internetski preglednik
- Jednostavno održavanje i nadogradnja na novu verziju

## ☐ Nedostatci

- Složenija izrada u odnosu na samostojne klijentske aplikacije
  - Potreba za posebno dizajniranim sučeljem (Web design)
  - Mogući problemi pri prikazu u različitim preglednicima
  - Potrebna prilagodba regionalnim posebnostima korisnika
- Sigurnosni problemi (neovlašten pristup aplikaciji i poslužitelju, zatrpavanje prometom)

# ASP.NET Web Forms

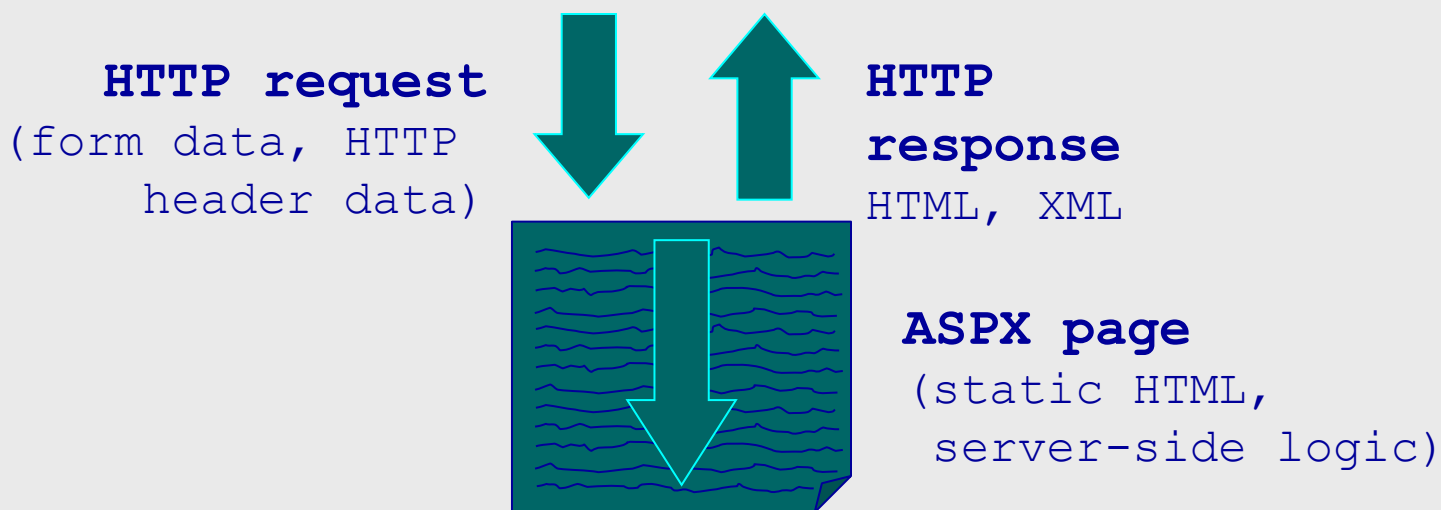
## ❑ ASP.NET - Active Server Pages .NET

- dinamičko generiranje stranica na poslužitelju
- povrh .NET Frameworka i IIS poslužitelja (Internet Information Services)

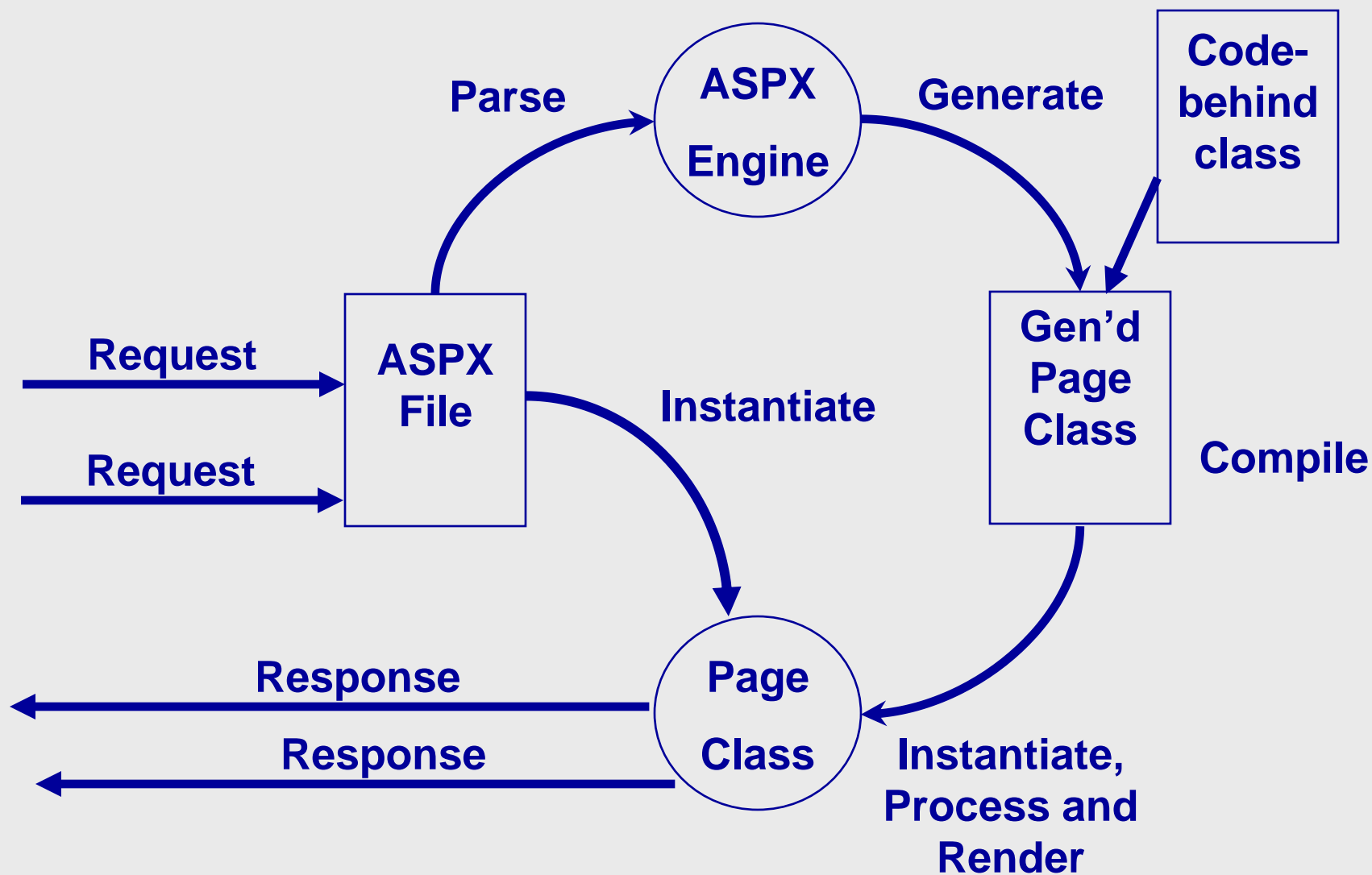
## ❑ Elementi aplikacija: Web Forms i Web controls

- poslužiteljska logika se piše u nekom od .NET jezika (C#, Visual Basic)
  - prevođenje → brže izvođenje nego kod skriptnih jezika (ASP, PHP)
- nalik windows formama (s ograničenjima)

## ❑ Alternativa unutar .NET-a - ASP.NET MVC



# Prevođenje i prikaz stranica



# Web Application ili Web Site?

## ❑ New \ Project \ Web \ ASP.NET Web Application – **koristiti ovo!**

- Stranica.aspx + Stranica.aspx.cs + Stranica.aspx.designer.cs (CodeBehind)
  - Automatsko dodavanje prostora imena pri kreiranju pojedine stranice/kontrole
  - Naziv razreda oblika WebAppName.Putanja.Stranica
- Eksplicitno dodavanje datoteka u projekt
- Jedan dll nakon Builda
- Kompilacija čitave aplikacije
- Mogućnost definiranja događa prije i poslije izgradnje aplikacije (PreBuild & PostBuild)

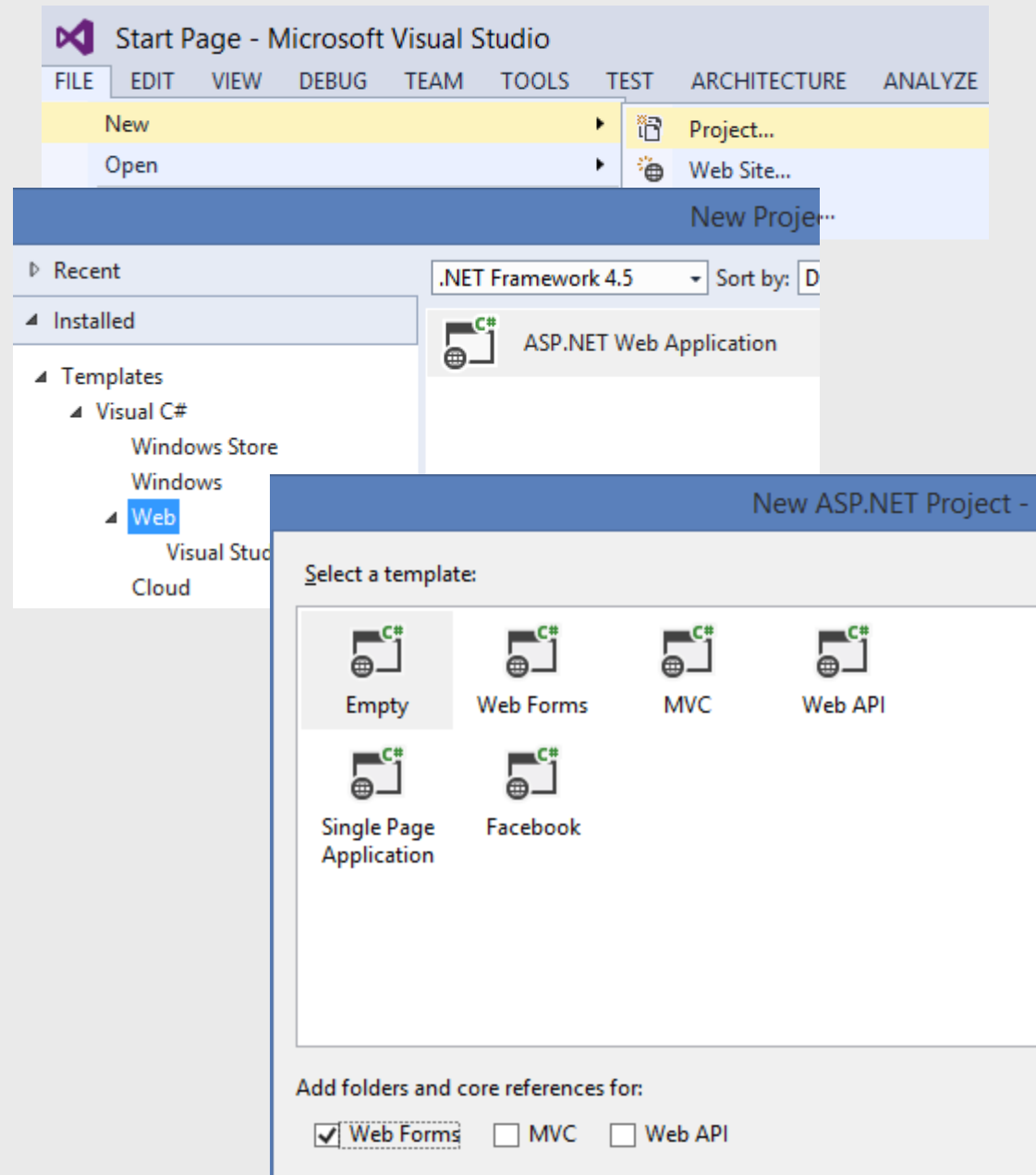
## ❑ New \ WebSite... \ ASP.NET Web Site

- Nije potreban project/solution
- Stranica.aspx + Stranica.aspx.cs (CodeFile)
  - Naziv razreda oblika Putanja\_Stranica
- Sve datoteke unutar Web mjesta su automatski dio “projekta”
- Datoteke s razredima nevezanim za pojedinu stranicu smještaju se u mapi App\_Code
- “In Place” kompilacija
- App\_Code.dll + po jedan dll za svaku stranicu
  - VS 2012/2013 omogućava promjenu postavki tako da se dobije samo jedan dll proizvoljnog imena

# Izrada web aplikacije

## □ New \ Project...

- ASP.NET Web Application
- Odabrati predložak Empty
- Označiti Web Forms



# Web poslužitelj

- ❑ **Gotova aplikacija treba biti postavljena na neki poslužitelj**
- ❑ **Primjer: Internet Information Services (IIS) – Microsoft web poslužitelj**
  - **Desni klik na web aplikaciju -> Publish Web Site**
    - Opcijionalno se objavljuju samo datoteke potrebne za izvršavanje aplikacije: \*.aspx, \*.ascx, web.config, bin/\* (bez izvornog koda)
- ❑ **IIS Express**
  - Privremeni web poslužitelj – pokreće se pokretanjem aplikacije u Visual Studiu
  - Pokrenut je dok i aplikacija
  - Ne može mu se pristupiti osim s lokalnog računala
- ❑ **Podrazumijevana stranica**
  - Default.aspx stranica (najčešće i Default.html, Default.php)
  - Može se promijeniti (Desni klik na stranicu -> Set as Start Page)



# Posebne mape web aplikacije

❑ Desni klik na web aplikaciju → Add ASP.NET Folder

Mapa	Značenje
App_Browsers	Sadrži datoteke kojima se identificira pojedini preglednik i utvrđuju njegove mogućnosti
App_Code	Pomoćni razredi i kod koji nije vezan isključivo za jednu stranicu
App_Data	Mapa za spremišta podataka (Access, SQL Express, XML datoteke, ...)
App_GlobalResources	Sadrži *.resx datoteke s podacima koji se mogu dohvatiti programski (najčešće tekstovi za različite jezike)
App_LocalResources	Sadrži *.resx datoteke vezane za pojedinu stranicu
App_Themes	Sadrži datoteke kojim se definiraju teme (izgled) web stranica i web kontrola
App_WebReferences	Mapa za sheme i kosture razreda za rad s pojedinim web servisom
Bin	Sadrži dll datoteke. Sadržaj ove mape automatski je referenciran unutar projekta

# Web Forms

## ❑ Web Forms - forme za izradu Web aplikacija

- Web stranica je objekt, izveden iz `System.Web.UI.Page`

## ❑ Razvoj stranica korištenjem kontrola, slično Windows Forms

- kontrole su objekti, izvedeni iz `System.Web.UI.Control`
- za svaku kontrolu postoji zasebni razred
  - konzistentniji objektni model, bolja provjera tipova pri prevođenju
- Obrada događaja standardno se obavlja na poslužitelju

## ❑ Realizacija

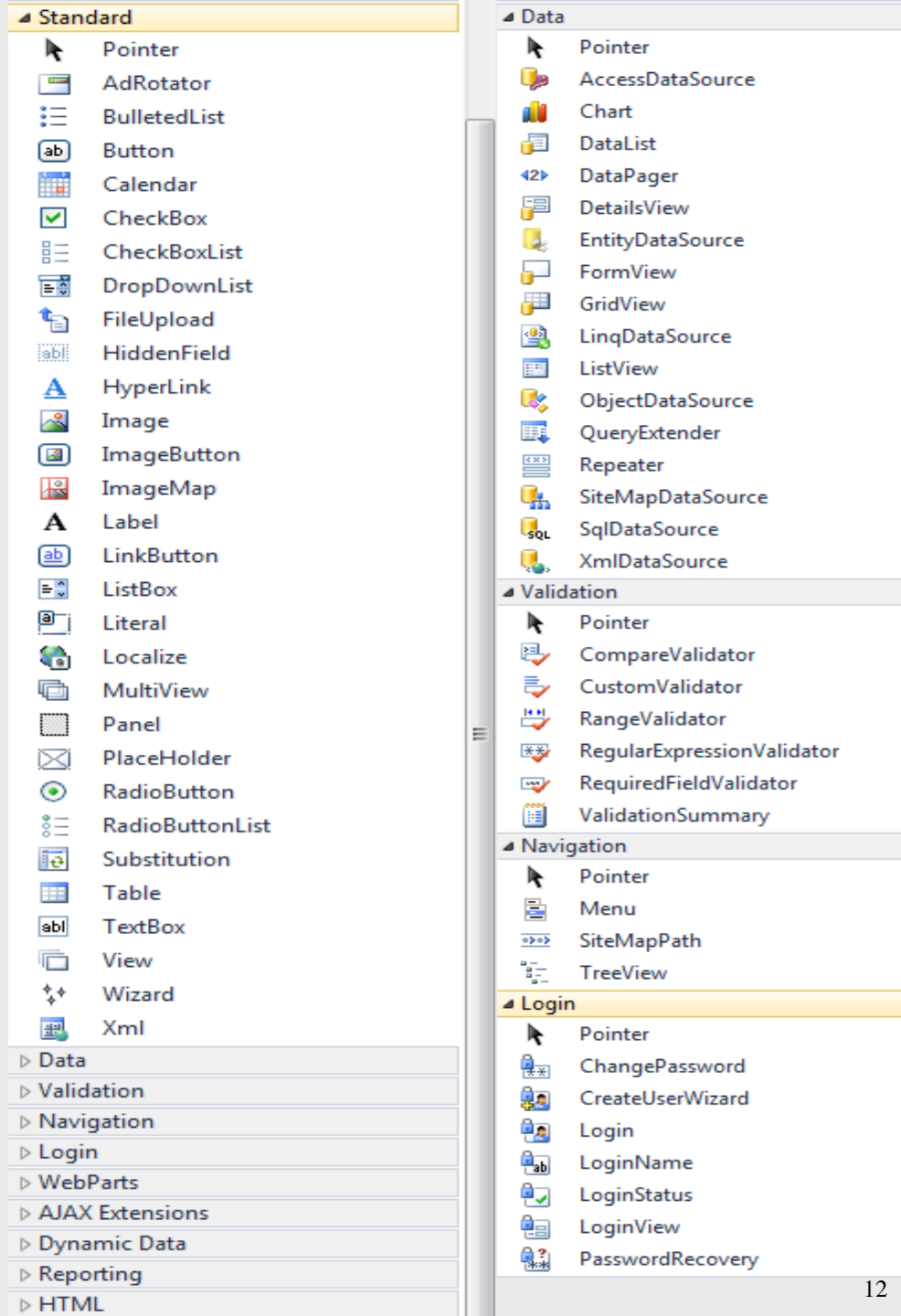
- .NET Framework vrši transformaciju kontrola u HTML
- kontrole imaju mogućnosti HTML kontrola uz dodatnu funkcionalnost
  - npr. `AutoPostBack`, validacijske kontrole, dodatni postupci
- `PostBack` – proces podnošenja stranice poslužitelju na obradu (promjene na stranici šalju se poslužitelju koji obrađenu stranicu vraća pregledniku)
- Da bi se očuvao sadržaj stranice, podaci se prenose zajedno sa stranicom
- Za rekonstrukciju sadržaja koriste se skrivena polja (`__VIEWSTATE`)

# Uobičajeni postupak izrade web stranica

1. Izbor željenih kontrola (Label, DropDownList, GridView)
  2. Imenovanje kontrola koje referenciramo u kôdu
  3. Dizajn forme - razmještaj kontrola i postavljanje svojstava
  4. Programiranje događaja na poslužitelju
- ☐ Uobičajeni redoslijed događaja (**poslužitelj**, **mreža**, **klijent**):
1. **zahtjev za stranicom se šalje poslužitelju**
  2. **Page\_Load (IsPostBack je False)**
  3. **stranica se šalje klijentu**
  4. prikaz stranice u pregledniku
  5. akcija (npr. klik na Button1) aktivira slanje
  6. **zahtjev za stranicom se šalje poslužitelju (zajedno s podacima)**
  7. **Page\_Load (IsPostBack je True)**
  8. **obrada događaja kontrole (npr. Button1\_Click, Textbox1\_Changed)**
  9. **stranica se šalje klijentu**
  10. prikaz stranice u pregledniku

# Serverske kontrole

- ❑ ASP.NET sadrži šezdesetak unaprijed definiranih serverskih kontrola podijeljenih u nekoliko grupa



# Osnovne Web kontrole

## ❑ System.Web.UI.WebControls

### ❑ Web kontrole za unos podataka s forme:

- TextBox (generirat će HTML nalik na `<INPUT TYPE="text">`)
- Button (`<INPUT TYPE="submit">`)
- CheckBox (`<INPUT TYPE="checkbox">`)
- DropDownList (`<SELECT>...</SELECT>`)
- ...

### ❑ Web kontrole koje omogućavaju povezivanje s izvorom podataka:

- GridView
- DataList
- Repeater
- FormView
- ...

### ❑ Ostale Web kontrole:

- Label (običan tekst)
- HyperLink (`<A HREF="URL"> VidljiviTekst </A>`)
- ...

# Sintaksa kontrola

## ❑ Kontrole se definiraju kao XML oznake, slično kao HTML oznake (tags)

- `<Kontrola id="naziv" runat="server" [attribute(s)]></Kontrola>`

```
<asp:Button id="Button1" runat="server" Text="Pošalji"></asp:Button>
```

- `<Kontrola id="naziv" runat="server" [attribute(s)] />`

```
<asp:TextBox id="Textbox1" runat="server" />
```

## ❑ Kontrole su implementirane ASP.NET razredima

- za imenovanje kontrole koristi se atribut `id` uz oznaku da se kontrola generira na serveru `runat="server"`
- oznake i atributi razlikuju velika i mala slova

## ❑ Svojstva kontrola mogu se postavljati programski:

```
txtCijena.Text="100";
```

## ❑ Obrada događaja obavlja se pozadinskim kodom (primjer slijedi...)

# Sintaksa aspx stranice

## ❑ Osnova stranice je HTML tekst

- Bilo koja HTML stranica može biti prozvana .aspx

## ❑ Standardne HTML oznake: <oznaka> [</oznaka>]

- <html> - HTML dokument
- <head> - zaglavlje HTML dokumenta
- <title> - naslov HTML dokumenta
- <br> - skok u novi redak
- <p> - paragraf, odjeljak teksta
- <span> - generički spremnik

## ❑ Stranica sadrži specijalne ASP.NET oznake

- Direktive (Directives): <%@ Page Language="C#" %>
- Serverske kontrole: <asp:Button runat="server">
- Odsječke kôda: <script runat="server">...</script>
- Izraze za povezivanje podataka: <%# %>
- Komentare kôda koji se izvršava na serveru: <%-- --%>
- "Mješoviti" kôd (isprepleteni html i .NET kod) <%= %> i <% %>
  - naslijeđeno iz ASP-a i treba izbjegavati

# Neka svojstva stranice

## ❑ <%@ Page attribute="value" [attribute="value"...] %>

- `AspCompat` - kompatibilnost s ASP, "true" izaziva pad performanci
- `AutoEventWireup` – automatsko povezivanje kontrola s događajima
- `Buffer` - korištenje međuspremnik za HTTP odzive
- `CodeFile` – izvorna datoteka s pozadinskim kodom
  - pr. `CodeFile="Default.aspx.cs"`
- `CodePage` – kodna stranica
- `EnableViewState` – automatizirano održavanje sadržaja pri opetovanom zahtjevu za stranicom, standardno "true"
- `ErrorPage` – URL na koji se preusmjerava u slučaju neobrađene pogreške
- `Inherits` – osnovni razred Page objekta
- `Language` – programski jezik
  - pr. `<%@ Page language="c#"`
- `MasterPageFile` – putanja do predloška s izgledom
- `Trace` – aktivira praćenje izvođenja stranice

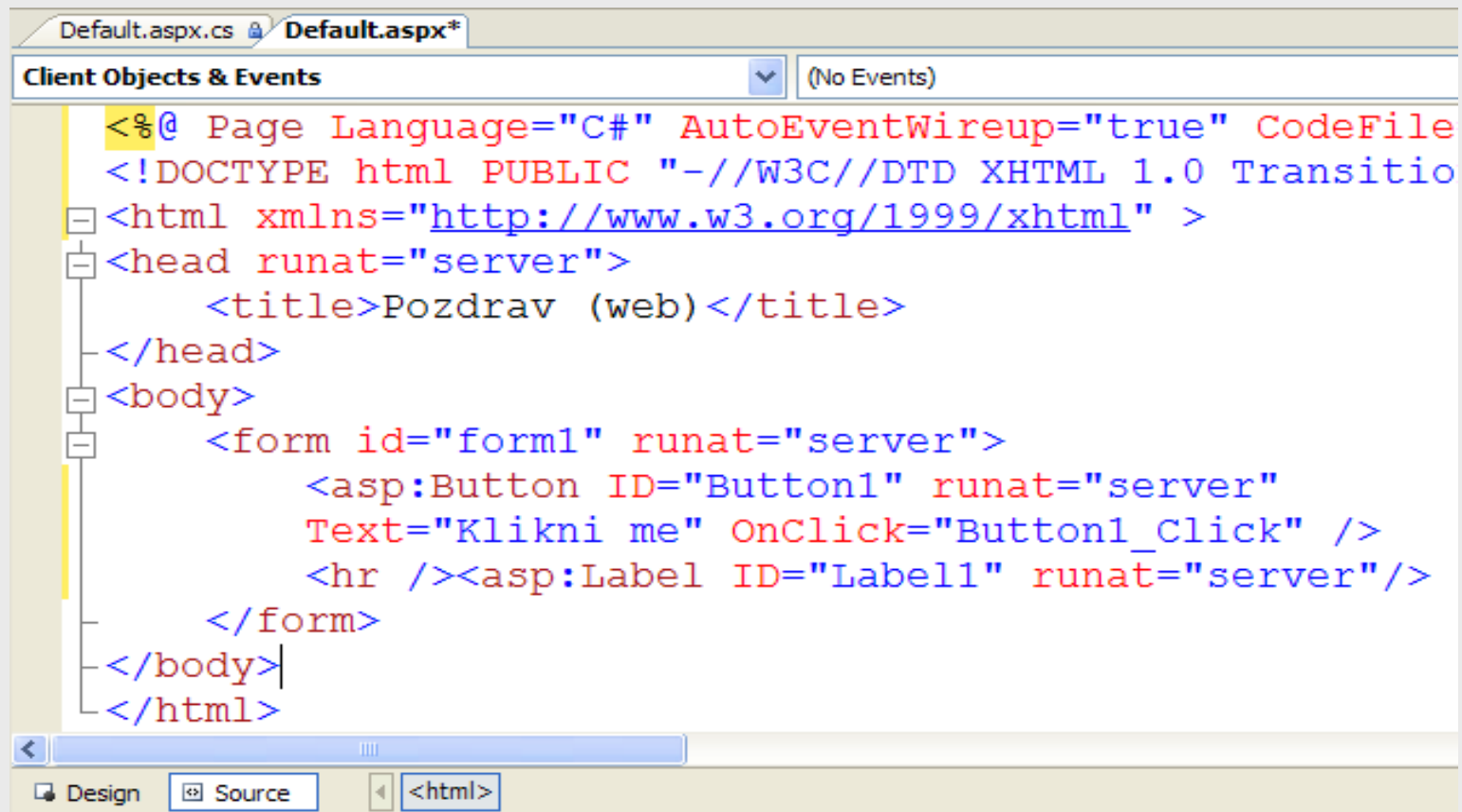
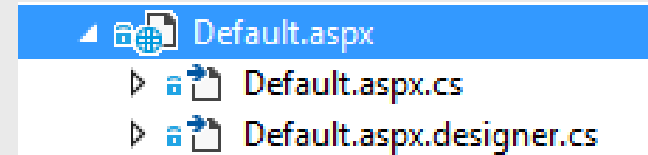
## ❑ Smije postojati samo jedna *Page* direktiva u .aspx datoteci



# Odvajanje dizajna od koda

## ❑ Datoteka s kontrolama - Default.aspx

- ASPX stranica uobičajeno sadrži HTML, serverske kontrole, JavaScript i ostale prezentacijske elemente
- Source i Design pogled na istu stranicu



# Odvajanje dizajna od koda

## ❑ Datoteka s kôdom (CodeFile, Code behind) – Default.aspx.cs i Default.aspx.designer.cs

- parcijalni razred
- sadrži kôd koji se izvršava na serveru

```
public partial class Default : System.Web.UI.Page {  
    protected void Page_Load(object sender, EventArgs e){  
    }  
    protected void Button1_Click(object sender, EventArgs e)  
    {  
        Label1.Text = "Pozdrav, trenutno vrijeme je " +  
                        DateTime.Now.ToString();  
    }  
}
```

# Životni ciklus web stranice

## ❑ Redoslijed događaja:

Događaj	Značenje
PreInit	Stvaranje kontrola, primjena tema, uspostava glavne (master) stranice, postavljanje korisničkih profila (jezik, ...)
Init	Inicijaliziranje vrijednosti svojstava kontrola na osnovu poslanih podataka ili podataka iz <i>ViewStatea</i>
Load	U ovom trenutku sve kontrole su inicijalizirane i vrijednosti povezane => piše se kod po želji
Obrada konkretnog događaja	Obrada konkretnog događaja koji je uzrokovao <i>PostBack</i> (npr. klik na gumb, promjena označene vrijednosti u listi...)
PreRender	Iscrtavanje/ispisivanje pojedine kontrole na izlazni tok (HTTP response)
Unload	Uništavanje objekata
Error	Događaj se može dogoditi bilo kada tijekom životnog ciklusa

## ❑ Presretanje događaja u postupku oblika **Page\_Događaj** npr. **Page\_Init(object sender, EventArgs e)** (ako je za stranicu postavljeno `AutoEventWireup="true"`)

# Objekti ASP.NET aplikacije

❑ **Svaka Web aplikacija nasljeđuje `System.Web.HttpApplication`**

❑ **Neki članovi `System.Web.HttpApplication`**

- `Application` – stanje Web aplikacije (npr. `Application["Naslov"]`)
- `Request` – HTTP zahtjev za stranicom
- `Response` – odgovor na HTTP zahtjev (npr. `Response.Write("<br>")`)
- `Server` – računalo poslužitelj
- `Session` – posjet Web aplikaciji od strane istog korisnika (npr. `Session["Ime"]`)
- `User` – podaci o korisniku

❑ **Važniji događaji koji se mogu obraditi u `Global.asax`:**

- `Application_Start` – kada se dohvati prva stranica Web aplikacije
- `Application_End` – kada dolazi do gašenja aplikacije (npr. kad se web server zaustavlja)
- `Session_OnStart` – kada korisnik prvi puta pristupi Web aplikaciji
- `Session_End` – korisnik se odjavio ili je isteklo dodijeljeno mu vrijeme

# Glavna stranica i izbornici

---

Primjer:  RPPP-primjeri\FirmaWebForms


# Glavna stranica

## ❑ Glavna stranica (Master) - ekstenzija *.master*

- Predstavlja zajednički izgled (okvir, dizajn) za više stranica
- Uobičajeno sadrži zajedničke elemente (izbornike, zaglavlje, podnožje, ...)
- Web aplikacija može imati više master stranica
- Može sadržavati što i obična stranica uz jedan ili više okvira (ContentPlaceHolder)

## ❑ Primjer: FirmaWebForms – Site.master

```
<asp:ContentPlaceHolder ID="Content" runat="server">
```

- Sadržaji okvira će se puniti u pojedinoj stranici
- Stranica koja ima *master* stranicu, vlastiti sadržaj dodaje samo unutar okvira definiranih u master stranici
- Primjer:  FirmaWebForms – Artikli.aspx


```
<%@ Page Language="C#" MasterPageFile="~/Site.Master" ...%>  
<asp:Content ID="Content2"  
    ContentPlaceHolderID="Content" runat="server">...  
</asp:Content>
```

- Add New Item → Web Forms Master Page i Web Form with Master Page

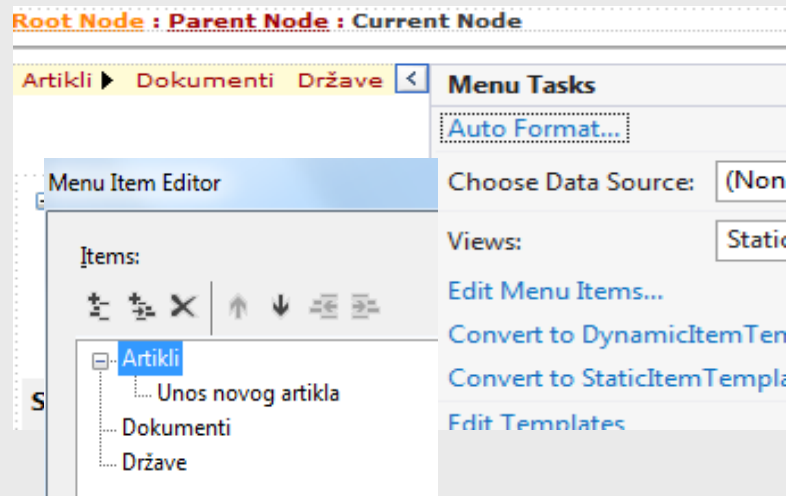
# Izbornici

- ❑ **Svaki izbornik je poveznica na određenu stranicu**
- ❑ **Dvije vrste izbornika:**
  - Prikaz izbornika slično kao kod Windows aplikacija (dinamički se prikazuju iz korijena izbornika) – kontrola *Menu*
  - Prikaz u obliku stabla - kontrola *TreeView*
- ❑ **Elementi izbornika mogu se navoditi:**
  - direktno unutar kontrole za navigaciju
  - definiranjem izvora podataka koji se veže na određenu mapu web-sjedišta
- ❑ **Trenutni položaj određene stranice i veza prema nadređenim stranicama može se prikazati kontrolom *SiteMapPath***
  - Za korištenje kontrole *SiteMapPath* potrebno je imati definiran izvor podataka za navigaciju, odnosno mapu web sjedišta.
- ❑ **Mapa web sjedišta je datoteka s ekstenzijom *.sitemap***
  - *Add New Item* → *Site Map*
  - Sadrži hijerarhijski popis stranica uključenih u izbornik navigacije
  - Pretpostavljeni naziv mape web aplikacije je *Web.sitemap*

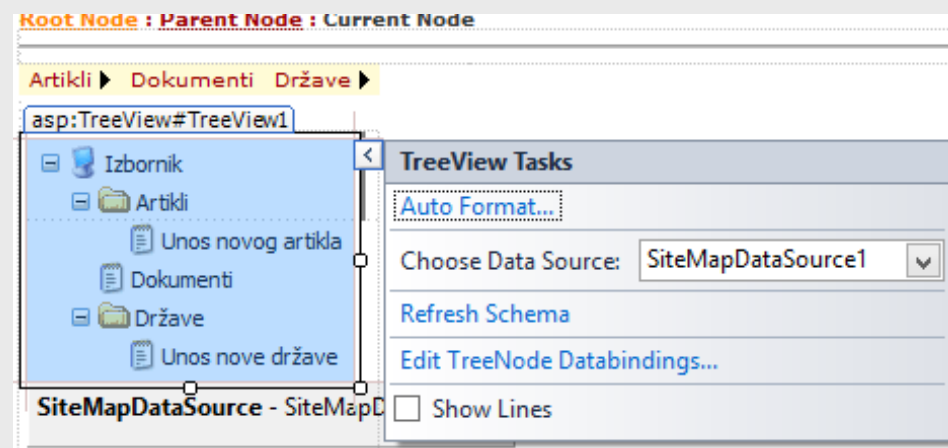
# Kontrola *Menu* – važnija svojstva

- ❑ **Orientation** – orijentacija prikaza prvog nivoa izbornika (horizontalno/vertikalno).
- ❑ **Items** – definira elemente (*MenuItem*) u izborniku
  - Elementi mogu biti ugniježđeni
- ❑ **DataSourceID** – izvor podataka (ako se ne koristi svojstvo **Items**)
- ❑ **Primjer:**  **FirmaWebForms – Site.master**
  - Kontrola **SiteMapPath** za prikaz trenutne pozicije + dva izbornika:

*Menu* s direktno navedenim vrijednostima



*TreeView* s podacima iz izvora podataka





# Definiranje mape web sjedišta

## ❑ Primjer: FirmaWebForms – Web.sitemap

- Xml datoteka sa popisom elemenata za izbornik
- Element *siteMap* sadrži elemente *siteMapNode* s atributima *url* i *title*

```
<siteMap xmlns=...>
  <siteMapNode url="" title="Izbornik">
    <siteMapNode url="~/Artikli.aspx" title="Artikli" >
      <siteMapNode url="~/UnosArtikla.aspx"
        title="Unos novog artikla" />
    </siteMapNode>
    <siteMapNode url="~/DokumentStavka.aspx"
      title="Dokumenti" />
    ...
  </siteMapNode>
</siteMap>
```

- Primjer povezivanja: *TreeView* na *Site.master* stranici

```
<asp:TreeView ID="TreeView1" runat="server"
  DataSourceID="SiteMapDataSource1">... </asp:TreeView>
<asp:SiteMapDataSource ID="SiteMapDataSource1" .../>
```

# Definiranje više mapa unutar web sjedišta

- ❑ Web sjedište može imati više mapa
- ❑ Nije nužno imati mapu *web.sitemap*
- ❑ U datoteci *web.config* mogu se definirati parametri za mape

```
<system.web>
  <siteMap defaultProvider="GlavniIzbornik">
    <providers>
      <add name="GlavniIzbornik"
            type="System.Web.XmlSiteMapProvider"
            siteMapFile="~/Navigacija/glavni.sitemap"/>
      <add name="Izbornik2"
            type="System.Web.XmlSiteMapProvider"
            siteMapFile="~/Navigacija/izbornik2.sitemap"/>
    </providers>
  </siteMap>      ...
```

- ❑ Prilikom definiranja izvora podataka za mapu potrebno je navesti pružatelja usluge za mapu web sjedišta

```
<asp:SiteMapDataSource ID="Izbornik2SiteMapDataSource"
  runat="server"
  SiteMapProvider="Izbornik2" />
```

# **Složene kontrole i povezivanje podataka**

---

**ObjectDataSource, GridView, FormView,  
DetailsView**

# Povezivanje podataka

## ❑ Povezani podatak na aspx stranici se prikazuje izrazom oblika `<%# expression %>`

- vezan za neki izraz ili vrijednost neke druge kontrolu

```
<asp:Label id="Label1" runat="server" Text="<%# 2+3 %>" />
```

```
<asp:label id="Label2" runat="server" Text="<%# TextBox1.Text %>" />
```

- vezan za neki izvor podataka

- Koriste se izrazi oblika `<%# Bind(...) %>` i `<%# Eval(...) %>` (objašnjeno naknadno)

## ❑ Izvor podataka - bilo koji razred koji implementira *IEnumerable* ili *IListSource*

## ❑ Načini povezivanja kontrole s izvorom podataka

- Programski u \*.aspx.cs datoteci

- pridruživanjem konkretnih podataka svojstvu *DataSource* i pozivom postupka *DataBind* na kontroli ili čitavoj stranici (povezuje sve kontrole sa stranice)

- U dizajnu

- postavljanjem svojstva *DataSourceId* na neku od posebnim kontrola za izvore podataka, npr. tipa *ObjectDataSource*, *EntityDataSource*, ...

# Povezivanje jednostavnih lista

- ❑ **Pojedini element sadrži vidljivi tekst i vrijednost koja se ne vidi.**
  - Npr. šifra artikla kao vrijednost, a naziv artikla kao tekst u padajućoj listi
  - Pod `DataValueField` navodi se što će se koristiti za vrijednost
  - Pod `DataTextField` navodi se što će se koristiti za tekst
  - Primjer slijedi naknadno (Slajd: *Odabir vrijednosti stranog ključa padajućom listom*)
  
- ❑ **Kontrole koje se povezuju na ovaj način**
  - `<asp:ListBox>`
  - `<asp:DropDownList>`
  - `<asp:RadioButtonList>`
  - `<asp:CheckBoxList>`

# Kontrola *ObjectDataSource*

## ☐ Služi za povezivanje podataka (slično kao *BindingSource*)

### ■ Važnija svojstva

- ☐ **DataObjectName** – tip podatka koji se povezuje
- ☐ **TypeName** – razred s postupcima za dohvat, brisanje, ažuriranje i sl.
- ☐ **SelectMethod** – naziv postupka za dohvat podataka
- ☐ **InsertMethod** – naziv postupka za dodavanje novog podatka
- ☐ **UpdateMethod** – naziv postupka za ažuriranje podatka
- ☐ **DeleteMethod** – naziv postupka za brisanje podatka
- ☐ **SelectCountMethod** – naziv postupka za ukupni broj elemenata
  - ☐ koristi se za izračun broja stranica
  - ☐ kad nije definiran vrši se dohvat svih (**SelectMethod**)
- ☐ **SortParamName** – naziv argumenta u postupku dohvata (**SelectMethod**) koji određuje redoslijed sortiranja
  - ☐ prazno ako izvor ne podržava sortiranje
  - ☐ nazivu svojstva se pridodaje sufiks DESC za silazno sortiranje
- ☐ **EnablePaging** – označava podržava li **SelectMethod** straničenje
  - ☐ **StartRowIndexParameterName** – naziv argumenta za poziciju početnog elementa (pretpostavljeno *startRowIndex*)
  - ☐ **MaximumRowsParameterName** – naziv argumenta za broj podataka koje treba dohvatiti (pretpostavljeno *maximumRows*)

# Definiranje postupaka za *ObjectDataSource* (1)






## ❑ Definira se:

- tip podatka (*DataObjectName*)
- razred za dohvat (*TypeName*)
- postupci za:
  - dohvat (*SelectMethod*),
  - dodavanje (*InsertMethod*)
  - ažuriranje (*UpdateMethod*)
  - brisanje (*DeleteMethod*)

## ❑ Parametar postupka je jednak tipu navedenom pod *DataObjectName* ili postoji više parametara koji imaju iste nazive kao nazivi povezanih stupaca

## ❑ U primjeru su implementirani pomoćni razredi (s navedenim postupcima) za rad s poslovnim objektima iz *Firma.BLL*

- Uključiti u projekt *Firma.BLL.dll*, *Firma.EF.dll* i *Firma.Framework.dll*
- *Web.config* sa *FirmaEntities* connection stringom

Properties	
ArtikliDataSource System.Web.UI.WebControls.ObjectDa	
    	
<b>DataObjectName</b>	<b>Firma.BusinessEntities.Artikl</b>
DeleteMethod	ObrisiArtikl
DeleteParameters	(Collection)
EnableCaching	False
EnablePaging	False
EnableViewState	True
FilterExpression	
FilterParameters	(Collection)
InsertMethod	DodajArtikl
InsertParameters	(Collection)
MaximumRowsParameter	maximumRows
OldValuesParameterFor	{0}
ParsingCulture	Invariant
SelectCountMethod	
SelectMethod	DohvatiSveArtikle
SelectParameters	(Collection)
SortParameterName	
SqlCacheDependency	
StartRowIndexParameter	startRowIndex
TypeName	ArtiklController
UpdateMethod	AzurirajArtikl

# Definiranje postupaka za *ObjectDataSource* (2)

## ❑ Primjer: FirmaWebForms – DataControllers\ArtiklController.cs

- Pomoćni razred za rad s poslovnim objektima
- Postupci primaju objekta tipa artikl (isti mora biti definiran kao DataObjectName)

```
[DataObject]
public class ArtiklController{
    [DataObjectMethod(DataObjectMethodType.Select, true)]
    public BusinessBaseList<Artikl> DohvatiSveArtikle() {
        ArtiklBllProvider bll = new ArtiklBllProvider();
        return bll.FetchAll();
    }
    [DataObjectMethod(DataObjectMethodType.Select, true)]
    public BusinessBaseList<Artikl> DohvatiArtikle(int startRowIndex,
                                                    int maximumRows) { ... }
    public void AzurirajArtikl(Artikl a){ ... }
    public void ObrisiArtikl(Artikl a){ ... }
    public void DodajArtikl(Artikl artikl){ ... }
```

- Čarobnjak za povezivanje izvora podataka i metoda za rad s podacima bit će jednostavniji ukoliko se iznad postupka stavi posebni atribut DataObjectMethod koji sugerira što postupak radi (Delete, Fill, Insert, Select, Update)
- Iznad razreda se stavlja atribut DataObject



# Povezivanje na *ObjectDataSource*

## ❑ Izvor podataka povezan na pomoćni razred: FirmaWebForms – Artikli.aspx

- 1. način – dohvatiti sve podatke sa izvora

```
<asp:ObjectDataSource ID="ArtikliDataSource" runat="server"  
    DataObjectName="Firma.BusinessEntities.Artikl"  
    TypeName="ArtiklController"  
    DeleteMethod="ObrisiArtikl" InsertMethod="DodajArtikl"  
    SelectMethod="DohvatiSveArtikle" UpdateMethod="AzurirajArtikl"/>
```

- 2. način (bolji) – dohvaćati samo potrebne podatke
  - Treba implementirati vlastito straničenje i sortiranje
  - Za straničenje treba definirati SelectCountMethod (ArtiklCount)

```
<asp:ObjectDataSource ID="ArtikliDataSource" runat="server"  
    DataObjectName="Firma.BusinessEntities.Artikl"  
    TypeName="ArtiklController"  
    DeleteMethod="ObrisiArtikl" InsertMethod="DodajArtikl"  
    SelectMethod="DohvatiArtikle" UpdateMethod="AzurirajArtikl"  
    EnablePaging="True" SelectCountMethod="ArtiklCount" />
```

- Slično, za sortiranje treba definirati SortParameterName (sortParam u primjeru DrzavaController)

# Kontrola *GridView*

## ❑ Mreža s podacima

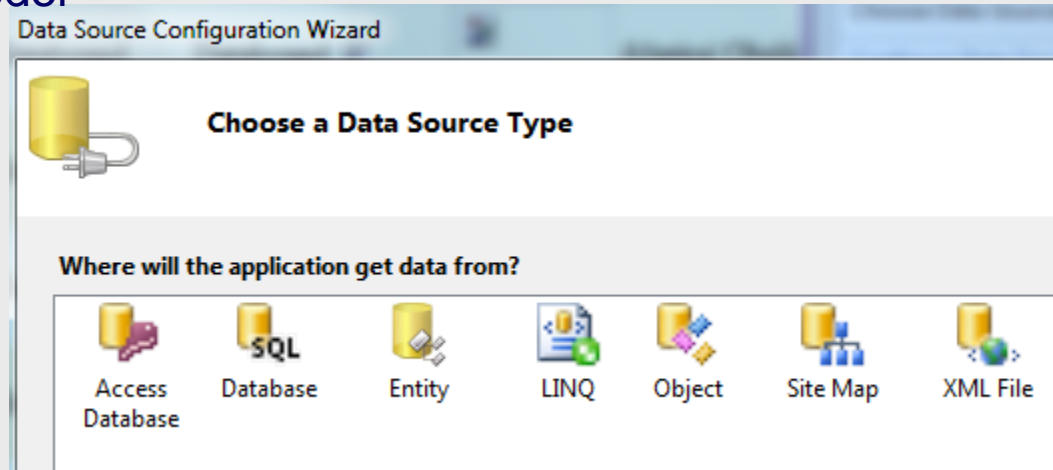
- Standardno prikazuje sve stupce (polja) povezanog izvora
  - može se definirati i podskup stupaca
  - pojedinačni stupac oblikuje se korištenjem predložaka (*templates*)
- mogućnost sortiranja podataka, pregleda po stranicama te izmjene i brisanja prikazanih podataka

## ❑ Vrste izvora podataka (svojstvo *DataSource*)

- Baza podataka podržana kroz ADO.NET
- Entity Framework ili LINQ model
- Poslovni objekt
- Mapa web-sjedišta
- XML datoteka
- Access

## ❑ Definiranje izvora podataka:

- U dizajnu
- Programski (tijekom izvođenja) nakon čega slijedi poziv metode `DataBind`
  - `DataBind` nije potreban ukoliko je unutar aspx stranice definiran `DataSourceId`



# Mreža s podacima – glavna svojstva

## ❑ Visual Studio – grafičko uređivanje mreže s podacima

- AutoFormat (odabir između nekoliko unaprijed definiranih izgleda mreže)
- EditColumns (grafičko sučelje za definiranje stupaca mreže)

## ❑ Važnija svojstva

- DataSourceId – izvor podataka kojim će se mreža popuniti
- AutoGenerateColumns – automatski prikaz svih stupaca iz povezanog izvora
- AllowSorting, AllowPaging – da li će se prikazati kontrole za sortiranje i straničenje
  - Straničenje je najčešće automatski podržano
    - Poželjno napisati postupak koji vraća n elemenata od neke pozicije
    - Ako takav postupak ne postoji uzimaju se svi podaci, pa se višak odbacuje
  - Sortiranje ovisi o izvoru podataka
    - Kad je izvor *ObjectDataSource* postupak za dohvat prima parametar za sortiranje (Primjer: `FirmaWebForms-DataControllers\DrzavaControler.cs`)
- DataKeyNames – kolekcija (primarnih) ključeva, omogućava jednoznačno određivanje pojedinog retka mreže prilikom brisanja i ažuriranja
  - npr: primarni ključ u prikazu artikala je `IdArtikla`
  - pojedinom ključu unutar kôda se pristupa pomoću kolekcije `DataKeys`  
npr: `grid.DataKeys[index].Value`

# Događaji unutar mreže s podacima

## □ Događaji na mreži s podacima za koje se može napisati vlastiti rukovatelj događaja:

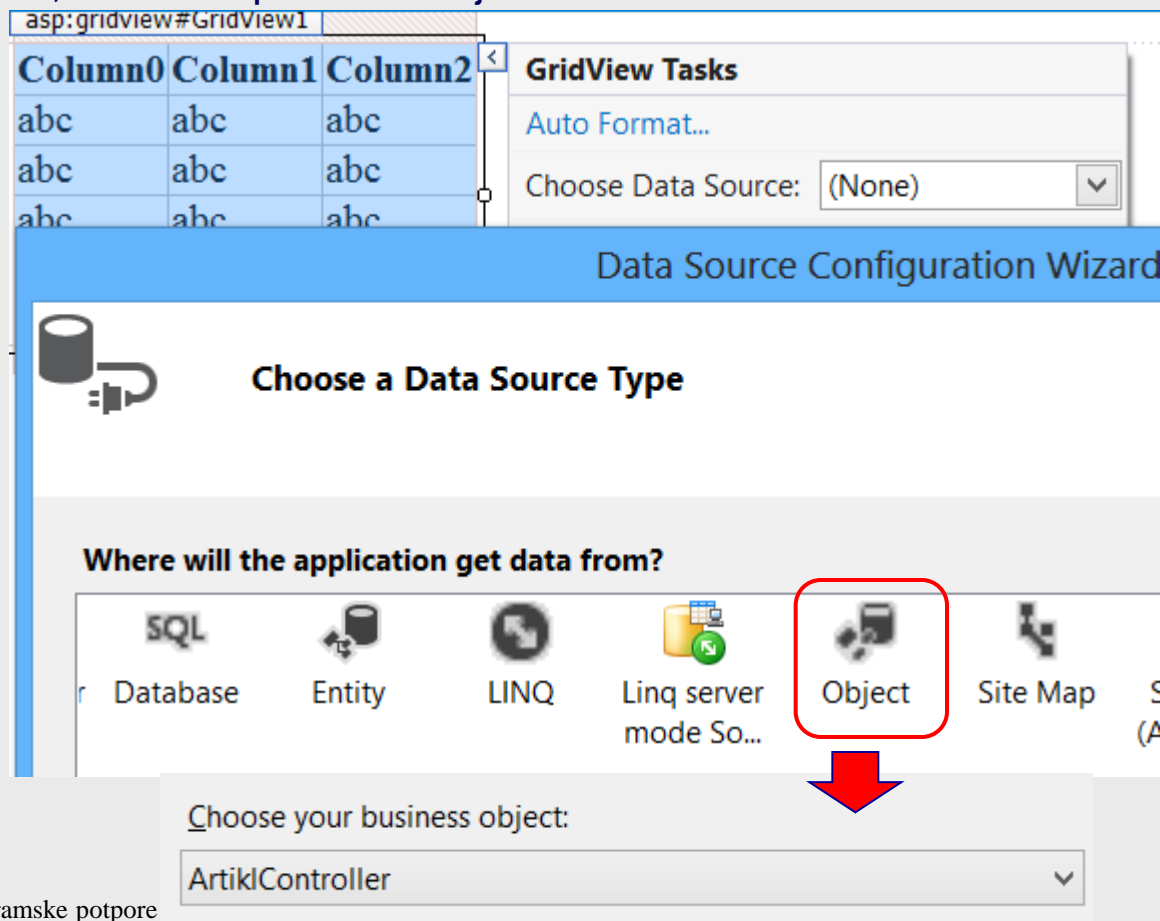
- DataBound – nastaje nakon što su podaci povezani s mrežom
- PageIndexChanged – nastaje nakon što je mreža s podacima obradila zahtjev za promjenu stranice
- PageIndexChanging – nastaje nakon klika na gumb za promjenu trenutne stranice unutar mreže s podacima, a prije nego mreža s podacima sama obradi događaj
- RowCancelingEdit – nastaje pri ažuriranju retka nakon klika na gumb `Cancel`, a prije nego se izađe iz načina rada koji omogućava ažuriranje
- RowCommand – nastaje nakon klika na gumb koji se nalazi unutar mreže s podacima
- RowCreated – nastaje nakon što se stvori pojedini redak u mreži
- RowDataBound – nastaje nakon što se pojedini redak poveže s podacima iz izvora
- RowDeleted – nastaje nakon brisanja pojedinog retka
- RowDeleting – nastaje pri brisanju retka, nakon klika na gumb `Delete`, a prije samog brisanja
- RowEditing – nastaje pri zahtjevu za ažuriranje retka mreže nakon klika na gumb `Edit`, a prije nego samog ulaza u način rada za ažuriranje
- RowUpdated – nastaje nakon ažuriranje pojedinog retka
- RowUpdating – nastaje nakon klika na gumb `Update`, ali prije samog ažuriranja
- SelectedIndexChanged – nastaje pri promjeni označenog retka
- SelectedIndexChanging – nastaje nakon klika na gumb `Select`, prije promjene označenog retka
- Sorted – nastaje nakon što je mreža s podacima sortirana po određenom stupcu
- Sorting – nastaje klikom na stupca po kojem se želi sortirati, a prije samog sortiranja
- DataBinding, Disposed, Init, Load, Unload, PreRender – naslijeđeno iz razreda `Control`

# Primjer dodavanja izvora podataka za GridView

❑ Primjer:  FirmaWebForms – Artikli.aspx

❑ Povezivanje se može izvesti u dizajnu

- klik na strelicu u gornjem desnom kutu iznad grida → *Choose Data Source* → *Object*
- Odabrati razred i postaviti postupke za dohvat, dodavanje, ažuriranje i brisanje
- Podesiti naslove stupaca, formate i poravnavanja



# Definiranje stupaca u mreži s podacima

- ❑ Potrebno je postaviti vrijednost svojstva `AutoGenerateColumns` na `false` (automatski odabirom konkretnog izvora podataka)
- ❑ Svojstvo `Columns` sadrži stupce koji mogu biti različitih tipova:
  - `BoundField` - vezani stupac. Povezuje se sa svojstvom iz izvora podataka navođenjem naziva u svojstvu `DataField`.
  - `TemplateField` - stupac definiran preko predložaka
  - `HyperLinkField` - stupac u kojem se nalaze poveznice
  - `CheckBoxField` - stupac u kojem su elementi tipa `CheckBox`
  - `ImageField` - stupac koji sadrži poveznicu na slike
  - `ButtonField` - stupac koji sadrži gumbе i predstavljaju proizvoljnu naredbu za mrežu s podacima. Gumb može biti prikazan kao poveznica koja izaziva `PostBack` (`LinkButton`), gumb u obliku slike (`ImageButton`) ili kao klasični gumb (`Button`).
  - `CommandField` - stupac čiji elementi mogu biti gumb za označavanje (`Select`), gumb za brisanje (`Delete`) i tri gumba za ažuriranje pojedinog retka (`Edit`, `Update`, `Cancel`). Tekst pojedinog gumba može se promijeniti
  - `DynamicField` – stupac koji koristi svojstva ASP.Net Dynamic-a

# Povezivanje pojedinačnih stupaca (1)

## ❑ Primjer: 📁 Web \ Firma.WebForms - Artikli.aspx

- Inicijalno svi stupci su vezani (BoundField) za određeno polje iz baze (DataField)

```
<asp:GridView ID="GridView1" runat="server"
    AutoGenerateColumns="False"
    DataKeyNames="SifArtikla" DataSourceID="ArtikliDataSource"
    PageSize="15" AllowPaging="True"
    onrowupdating="GridView1_RowUpdating">
    <Columns>
        <asp:BoundField DataField="SifArtikla" HeaderText="Šifra artikla"
            ReadOnly="True" />
        <asp:BoundField DataField="NazArtikla" HeaderText="Naziv" ... />
        ...
        <asp:BoundField DataField="CijArtikla" DataFormatString="{0:N2}"
            ...
    </Columns>
</asp:GridView>

<asp:ObjectDataSource ID="ArtikliDataSource" runat="server"
    ...
</asp:ObjectDataSource>
```

# Povezivanje pojedinačnih stupaca (2)

- ❑ **Ograničena funkcionalnost vezanih stupaca → predlošci za stupce**
  - Pri dizajnu stranice klik na gornji desni rub mreže s podacima → `Edit Columns` → klik na stupac → "Convert this field into a `TemplateField`"
- ❑ **Moguće definirati izgled:**
  - zaglavlja stupca (`HeaderTemplate`)
  - podnožja stupca (`FooterTemplate`)
  - pojedinog retka (`ItemTemplate`)
  - svakog drugog retka (`AlternatingItemTemplate`)
  - retka prilikom ažuriranja (`EditItemTemplate`)
  - novog retka u mreži (`InsertItemTemplate`)
- ❑ **Kod stupaca definiranih preko predložaka potrebno je eksplicitno povezati element stupca mreže sa stupcem u izvoru (*Eval*, *Bind*)**
  - Od verzije 4.5. umjesto *Eval* i *Bind* mogu se koristiti *Item* i *BindItem*
    - *Strongly typed* varijanta
    - Potrebno postaviti svojstvo *ItemType* na kontroli *GridView*



# Povezivanje pojedinačnih stupaca (3)

## ❑ **DataBinder – statički objekt za povezivanje**

## ❑ **Jednosmjerno povezivanje**

### ■ Postupak `Eval`

- prikaz i formatiranje podataka
- `<%# Eval("Cijena", "{0:N2}") %>`

### ■ Podacima se može pristupiti i drugačije u ovisnosti o tipu podataka koji se prikazuju pomoću mreže

- `DataTable`
  - `<%# ((DataRow)Container.DataItem) ["FieldName"] %>`
- `List<string>`
  - `<%# ((string)Container.DataItem) %>`

## ❑ **Dvosmjerno povezivanje**

### ■ Postupak `Bind`

- prikaz, ali i slanje vrijednosti kod *postbacka*
- koristi se npr. kod ažuriranja pojedinih redaka u mreži s podacima

### ■ `Bind`

- `<%# Bind("Cijena", "{0:N2}") %>`

# Ažuriranje redaka u mreži s podacima

- ❑ **"Gotova" funkcionalnost – za ažuriranje se brine izvor podataka**
  - Gumbi Edit / Cancel / Update
    - funkcionalnost prijelaza iz stanja prikaza u stanje ažuriranja
  - Ako je izvor podataka EntityFramework nije potrebno postavljati dodatne postavke
  - Ako je izvor podataka neki poslovni objekt, potrebno je definirati postupke za dohvat (`SelectMethod`), dodavanje (`InsertMethod`), ažuriranje (`UpdateMethod`), brisanje (`DeleteMethod`)
  
- ❑ **Automatsko povezivanje vrijednosti za vezane stupce i kontrole koje povezivanje ostvaruju s *Bind***
  - *ili BindItem od ASP.Net 4.5*

# Posebno dizajnirani stupci

## ❑ Primjer: FirmaWebForms – Artikli.aspx

- Limitiranost vezanih stupaca
  - koriste se stupci u obliku predloška (TemplateField)
- Stupac sa slikom prilagođen s posebnim izgledom za prikaz (ItemTemplate) i za ažuriranje (EditItemTemplate)

```
<Columns>
...
<asp:TemplateField>
    <EditItemTemplate>
        <asp:CheckBox ID="cbObrisiSliku" runat="server"
            Text="Obrisati sliku?" />
        <asp:FileUpload ID="UploadSlike" runat="server" />
    </EditItemTemplate>

    <ItemTemplate>
        <asp:Image ID="Image1" runat="server"
            Width="80px" Height="50px"
            ImageUrl='<%# Eval("SifArtikla",
                "SlikaArtikla.aspx?SifArtikla={0}") %>' />
    </ItemTemplate>
</asp:TemplateField>
...</Columns>
```

# Prikaz slike iz baze

- ❑ Primjer:  FirmaWebForms – SlikaArtikla.aspx
- ❑ Sadržaj stranice prazan, u cs-dijelu kod koji šalje sadržaj na izlazni tok

```
protected void Page_Load(object sender, EventArgs e) {  
    Response.ContentType = "img/jpeg";  
    int SifArtikla;  
    if (int.TryParse(Request["SifArtikla"], out SifArtikla)) {  
        Articl artikl = new ArticlBllProvider().Fetch(SifArtikla);  
        byte[] slika = artikl.SlikaArtikla;  
        if (slika != null)  
            Response.BinaryWrite(slika);  
        else  
            Response.WriteFile(Server.MapPath("~/bez slike.jpg"));  
    }  
}
```

- ❑ Bolje, ali složenije rješenje – koristiti *handler*e (razred koji implementira *IHttpHandler* + dodatno podešavanje u web.configu)

# Prijenos argumenata stranici

## ❑ Standardno URL je oblika

`http://server/putanja/stranica.aspx?naziv1=vrijednost1&naziv2=vrijednost2`

- Parametri iza upitnika čine tzv. *QueryString*
- Vrijednost pojedinog parametra iz URL-a može se dobiti pomoću `Request["Naziv parametra"]`

## ❑ Neka od svojstava klase Request:

- `UserHostAddress` : IP adresa klijenta (posjetitelja stranice)
- `UserAgent` : Internet preglednik klijenta
- `Request.Url.OriginalString`: Puni URL trenutne stranice
- `Request.Url.LocalPath` : Lokacija trenutne stranice na serveru (kazalo u kojem se aplikacija nalazi)

## ❑ Primjer: FirmaWebForms – SlikaArtikla.aspx

- Za direktan prikaz potrebno bi bilo dodati elemente HTML-a (kontrolu za prikaz slike)

# Povezivanje za posebno dizajnirane stupce

## ❑ Primjer: FirmaWebForms – Artikli.aspx

- Sva povezivanja ostvarena s *Bind* su automatska
- Ostale vrijednosti treba eksplicitno postaviti prilikom obrade pojedinog događaja
  - Obraduje se događaj *Updating* (početak ažuriranja određenog retka)
  - Vrijednosti za određeni ključ u kolekciji *e.NewValues*
  - Dohvat pojedine kontrole u retku vrši se postupkom *FindControl*
  - U primjeru se provjerava briše li se slika ili se uzima upravo poslana

```
protected void GridView1_RowUpdating(... GridViewUpdateEventArgs e) {  
    GridView grid = ((GridView) sender);  
    CheckBox cb = grid.Rows[e.RowIndex].FindControl("cbObrisiSliku")  
                as CheckBox;  
    if (cb.Checked)  
        e.NewValues["SlikaArtikla"] = null;  
    else  
    {  
        FileUpload fileUpload = grid.Rows[e.RowIndex]  
                                .FindControl("UploadSlike")  
                                as FileUpload;  
        if (fileUpload.FileBytes.Length > 0)  
            e.NewValues["SlikaArtikla"] = fileUpload.FileBytes;  
        ...  
    }
```

# ObjectDataSource i decimalna točka/zarez

## ❑ Primjer: FirmaWebForms – Artikli.aspx

- Zbog pogreške u kontroli *ObjectDataSource* vezano uz decimalnu točku/zarez, stupac za cijenu artikla izveden kao *TemplateField* bez korištenja *Bind*

```
<Columns>
...
<EditItemTemplate>
    <asp:TextBox ID="tbCijena" runat="server" Text='<%#
        Eval("CijArtikla", "{0:N2}") %>'></asp:TextBox>
</EditItemTemplate>
```

- Zahtjeva eksplicitno postavljanje vrijednosti u kodu (obrada događaja *RowUpdating* na gridu)
- Primjer:  FirmaWebForms – Artikli.aspx.cs


```
protected void GridView1_RowUpdating(...) {
    TextBox cijenaArtikla =
        grid.Rows[e.RowIndex].FindControl("tbCijena") as TextBox;
    e.NewValues["CijArtikla"] = decimal.Parse(cijenaArtikla.Text);
    ...
}
```

# Dodavanje novih zapisa

## ❑ **GridView** nema automatsku mogućnost dodavanja novih zapisa

- moguće definirati svaki stupac kao *TemplateField* i napraviti kontrole za dodavanje novog podatka u zaglavlju ili podnožju
  - Napravljeno za stavke u primjeru Dokument-Stavka
- Ponekad je lakše napraviti dodavanje na posebnoj stranici
  - Može se napraviti “pješice” (dohvatom pojedinačnih vrijednosti) ili korištenjem naprednijih kontrola: *FormView* ili *DetailView*

## ❑ **Primjeri:** **FirmaWebForms – UnosArtikla.aspx**

- *DetailsView* – prikaz pojedinačnog podatka s izvora u tabličnom obliku
- *FormView* – prikaz pojedinačnog podatka s izvora u proizvoljnom rasporedu
  - Iste vrste polja kao i kod *GridViewa*
- 3 načina prikaza (svojstva *DefaultMode* i *CurrentMode*):
  - *ReadOnly* – prikaz pojedinačnog podatka
  - *Insert* – prikazuju se polja za unos podataka i gumbi Insert i Cancel
  - *Edit* – prikazuju se polja za ažuriranje postojećih podataka i gumbi za Update i Cancel
- Detaljnije o FormViewu u primjeru za master-detail
- Sličan primjer, ali bez validacije:  **FirmaWebForms – NovaDrzava.aspx**



# Validacijske kontrole

## ❑ Primjer: FirmaWebForms – UnosArtikla.aspx

- Svaka validacijska kontrola odnosi se na neku kontrolu (svojstvo `ControlToValidate`) te ima tekst poruke za pogrešku (`ErrorMessage`)
- Provjera i prije slanja podataka na server (prije *postbacka*)
  - `RequiredFieldValidator`: provjera da li je određeno polje popunjeno
  - `RegularExpressionValidator`: provjera regularnog izraza definiranog u svojstvu `ValidationExpression`
- Provjera na serveru (nakon *postbacka*)
  - `CustomValidator`: navodi se postupak koji se treba izvesti na serveru (`onservervalidate`)

```
<asp:RegularExpressionValidator ... ErrorMessage="Cijena artikla  
mora biti decimalni broj s maksimalno dvije decimale"  
ValidationExpression="\d*(\,\d{1,2})?"  
ControlToValidate="tbCijenaArtikla" />
```

```
<asp:RequiredFieldValidator ... ControlToValidate="tbJedinicaMjere"  
ErrorMessage="Jedinica mjere mora biti unešena." />
```

```
<asp:CustomValidator ... ErrorMessage="Šifra artikla već postoji"  
onservervalidate="CustomValidatorSifArtikla_ServerValidate"  
ControlToValidate="tbSifArtikla"/>
```

# Izgled kontrola

## ❑ Izgled je u primjer zadan odvojeno od koda

- Cascading Style Sheets (**CSS**) datoteka definira stilove –  FirmaWebForms – Content\Site.css

## ❑ Primjena u kodu:

```
<asp:GridView ... AllowPaging="True"
    PageSize="15" CssClass="zebra" ... >
    ...
</asp:GridView>
```

## ❑ Alternativa:

```
<asp:GridView ...
BackColor="LightGoldenrodYellow"
BorderColor="Tan" BorderWidth="1px"
CellPadding="2" ForeColor="Black"
GridLines="None" ...>
<AlternatingRowStyle
BackColor="PaleGoldenrod" />
```

```
.zebra
{
border: thin solid
black;
padding: 2px;
border-spacing : 0px;
margin: 0px;
text-align: left;
}
```

```
.zebra tbody tr:nth-
child(2n+1) {
background-color:
lightGoldenrodYellow;
}
```

```
.zebra tbody tr:nth-
child(2n) {
background-color:
paleGoldenrod;
}
```

```
.zebra tbody tr th {
background-color:
tan;
}
```

# Primjer Dokument-Stavka

---

Primjer:  FirmaWebForms –  
DokumentStavka.aspx

# Web stranica oblika zaglavlje-stavke

## ❑ Primjer: FirmaWebForms – DokumentStavka.aspx

- pojedinačni dokument (unutar `FormView`) i njegove stavke unutar kontrole `GridView`
- Mogućnost izmjene podataka (dodavanje, brisanje, ažuriranje)
- Validacija podataka prilikom unosa

<b>IdDokumenta</b>	2457	<b>Vrsta</b>	O	<b>Broj</b>	48	<b>Datum</b>	3.1.2010
<b>Partner</b>	VENDOR (742708)					<b>Porez</b>	22,00%
<b>Prethodni dokument</b>	2454/3.1.2010 - VDU (80039202) : 155.225,11 kn					<b>Iznos</b>	220.075,67 kn
<input type="button" value="Ažuriraj"/> <input type="button" value="Obriši"/> <input type="button" value="Dodavanje novog dokumenta"/>							
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ... >>							

→ FormView

Artikl	Količina	Jedinična cijena	Rabat	Iznos	
USB punjač za PDA Handspring BANGRIDGE (CM 45032)	5,00000	62,00 kn	5,00%	294,50 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
Knjiga "Digitalni fotoaparat - ukratko"	3,00000	30,00 kn	1,00%	89,10 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
Docking station, univerzalni KINGSINGTON, USB (60113)	4,00000	399,00 kn	6,00%	1.500,24 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
Ruksak za notebook, PATACO MBP-1	4,00000	146,00 kn	9,00%	531,44 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
Glazbena linija, GROUNDIG Ovation 2 CDS 7000 DEC, MP3	50,00000	1.299,00 kn	5,00%	61.702,50 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
Mobilni uređaj SUNNY NICOLSON M600i, GPRS, IrDA, MMS, ekran u boji, crni	2,00000	1.889,00 kn	7,00%	3.513,54 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
MP3/WMA/FM/Video player, iAUDIO, i6, HDD 4 GB	6,00000	1.299,00 kn	2,00%	7.638,12 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
MP3/WMA/FM player, iAUDIO, T2, 2 GB	2,00000	799,00 kn	4,00%	1.534,08 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
MBO SLAFROCK, s. 775, 4COREDUAL-VSTA, VIA PT880 Ultra, BUS 1066 MHz, serial ATA, RAID, 7.1 zvuk, LAN, SSSR 2, ATX 2	4,00000	428,00 kn	2,00%	1.677,76 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
KPD Athlon 64 X2 4400+, 2300 MHz, 1 Mb c., socket AM2, BUS 2000 MHz, 65 nm, BOX (Brisbane Dual Core)	2,00000	557,00 kn	3,00%	1.080,58 kn	<a href="#">Ažuriraj</a> <a href="#">Obriši</a>
A/C kućni punjač za Palmu					<a href="#">Dodaj</a>

# Izvori podataka za primjer Dokument-Stavka

## □ Primjer: FirmaWebForms – DokumentStavka.aspx

- Za izvore korištena kontrola *ObjectDataSource*
- *DokumentDataSource*
  - izvor podataka za dokument
  - postupci u *DokumentControler.cs*
- *StavkeDataSource*
  - izvor podataka za stavke pojedinačnog dokumenta
  - postupci u *DokumentControler.cs*
- *ArtiklDataSource*
  - izvor podataka za odabir artikla iz padajuće liste za pojedinu stavku dokumenta
  - Postupak za dohvat definiran u *ArtiklLookupController.cs*
- *PrethodniDokumentDataSource*
  - izvor podataka za padajuću listu za odabir prethodnog dokumenta
  - Postupak *FetchLookup* iz *DokumentBIIProvidera* (iz primjera Firma.Win)
- *PartnerDataSource*
  - izvor podataka za padajuću listu za odabir partnera
  - Postupak *FetchLookup* iz *PartnerBIIProvidera* (iz primjera Firma.Win)

**ObjectDataSource** - DokumentDataSource

**ObjectDataSource** - StavkeDataSource

**ObjectDataSource** - PrethodniDokumentDataSource

**ObjectDataSource** - PartnerDataSource

**ObjectDataSource** - ArtiklDataSource

# Kontrola *FormView*

## ❑ Primjer: FirmaWebForms – DokumentStavka.aspx

- Omogućava prikaz jednog podatka u proizvoljnom rasporedu i podržava straničenje
- Određen izvorom podataka i primarnim ključem (može biti kompozitni)
  - DataSourceID="DokumentDataSource"
  - DataKeyNames="IdDokumenta"
- 3 načina prikaza (ReadOnly, Insert, Edit) – posebni predlošci

FormViewDokument - ItemTemplate

ItemTemplate

<b>IdDokumenta</b> [IdDokumentaLabel]	<b>Vrsta</b> [VrDokumentaLabel]	<b>Broj</b> [BrDokumentaLabel]	<b>Datum</b> [DatDokumentaLabel]
<b>Partner</b> [IdPartneraLabel]			<b>Porez</b> [PostoPorezLabel]
<b>Prethodni dokument</b> [Databound ▼]			<b>Iznos</b> [IznosDokumentaLabel]

Ažuriraj Obriši Dodavanje novog dokumenta

InsertItemTemplate

<b>Vrsta</b> [RegularExpressionValidatorVrsta] *	<b>Broj</b> [CustomValidatorBrDokumenta] [RegularExpressionValidatorBroj] *	<b>Datum</b> [RegularExpressionValidator1] *
<b>Partner</b> [Databound ▼]		<b>Porez</b> [RegularExpressionValidatorPorez] *
<b>Prethodni dokument</b> [Databound ▼]		<ul style="list-style-type: none"><li>• Error message 1.</li><li>• Error message 2.</li></ul>

Spremi Odustani







# Dohvat dokumenata

## ❑ Primjer: Web \ Firma.WebForms – AppCode\DokumentController.cs

- Postupak za dohvat ima atribut `DataObjectMethod`
  - Nije nužno, ne utječe na funkcionalnost
  - Olakšava odabir u čarobnjaku za definiranje izvora podataka
- Za dohvat se koristi `Firma.Bll.dll` iz primjera `Firma.Win`

```
[DataObject]
public class DokumentController {

    [DataObjectMethod(
        DataObjectMethodType.Select, true)]
    public DokumentList DohvatiDokumente() {
        var bll = new DokumentBllProvider();
        var list = bll.FetchLazy();
        return list;
    }
}
```

Properties  	
<b>DokumentDataSource</b> System.Web.UI.WebCont	
   	
DeleteMethod	<b>ObrisiDokument</b>
DeleteParameters	(Collection)
EnableCaching	False
EnablePaging	False
EnableViewState	True
FilterExpression	
FilterParameters	(Collection)
InsertMethod	<b>InsertDokument</b>
InsertParameters	(Collection)
MaximumRowsParam	maximumRows
OldValuesParameterF	{0}
SelectCountMethod	
SelectMethod	<b>DohvatiDokumente</b>
SelectParameters	(Collection)
SortParameterName	
SqlCacheDependency	
StartRowIndexParam	startRowIndex
TypeName	<b>DokumentController</b>
UpdateMethod	<b>AzurirajDokument</b>

# Predložak za prikaz

## ❑ Primjer: FirmaWebForms – DokumentStavka.aspx

- Dizajnira se unutar oznake *ItemTemplate* unutar *FormView*a
- Za prikaz pojedine vrijednosti iz dokumenta koristi se postupak *Eval* uz opcionalno navođenje formata prikaza
  - `<%# Eval("IdDokumenta") %>`
  - `<%# Eval("IznosDokumenta", "{0:C2}") %>`
  - `<%# Eval("DatDokumenta", "{0:d.M.yyyy}") %>`
- Na dnu predloška dodani gumbi za prelazak u neki od drugih načina prikaza (naredbe *Edit* i *New*) i gumb za brisanje pojedinog dokumenta (naredba *Delete*)
  - Nijedan od gumba ne izaziva validaciju (`CausesValidation=false`) - korisno u slučaju da se na stranici neki drugi podatak nalazi u stanju ažuriranja
  - Gumbu za brisanje pri inicijalizaciji (obrada događaja *Init*) se dodaje *javascript* kod za potvrdu brisanja (pogledati postupak `DeleteButton_Init`)

```
<asp:Button ID="EditButton" runat="server" Text="Ažuriraj"
CausesValidation="False" CommandName="Edit" />
<asp:Button ID="DeleteButton" runat="server" Text="Obriši"
CausesValidation="False" CommandName="Delete"
oninit="DeleteButton_Init"/>
<asp:Button ID="DodajButton" runat="server" CausesValidation="False"
CommandName="New" Text="Dodavanje novog dokumenta" />
```



# Predložak za ažuriranje dokumenta

## ❏ Primjer: FirmaWebForms – DokumentStavka.aspx


- Dizajnira se unutar oznake *EditItemTemplate* unutar *FormView*a
- Za podatke koji se mogu promijeniti koristi se kontrole *TextBox*, *DropDownList*, ... i povezivanje postupkom *Bind* uz opcionalno navođenje formata prikaza
  - `<asp:TextBox ID="tbDatDokumenta" runat="server" Text='<%# Bind("DatDokumenta", "{0:dd.MM.yyyy}") %>' />`
  - `<asp:DropDownList ID="ddlPartneri" runat="server" SelectedValue='<%# Bind("IdPartnera") %>' DataSourceID="PartnerDataSource" DataTextField="Text" DataValueField="Key" />`
- Sve vrijednosti povezane s *Bind* automatski se proslijeđuju izvoru
- Na dnu predloška dodan gumb za spremanje (naredba *Update*) i gumb za odustajanje (naredba *Cancel*)
  - Za sve kontrole za unos dodati validacijske kontrole i postaviti svojstvo *ValidationGroup* na vrijednost kao i kod gumba za spremanje promjena

```
<asp:Button ID="UpdateButton" runat="server" CausesValidation="True"
CommandName="Update" Text="Spremi" ValidationGroup="Dokument" />
```

```
<asp:Button ID="UpdateCancelButton" runat="server"
CausesValidation="False" CommandName="Cancel" Text="Odustani" />
```

# Odabir vrijednosti stranog ključa padajućom listom (1)

## ❑ Primjer: FirmaWebForms – DokumentStavka.aspx

- Partner, prethodni dokument i artikl u stavkama odabiru se iz padajuće liste
- Izvori tipa *ObjectDataSource* vežu se na *BIProvidere* iz poslovnog sloja
- Za dohvat se koristi postupak *FetchLookup*
  - Iznimno, zbog performansi padajuća lista za artikle spremljena u memoriju, pa se koristi pomoćni razred
    - Primjer:  FirmaWebForms – DataControllers\ArtiklLookupController.cs
- *FetchLookup* vraća listu elemenata tipa *LookupData* koji ima *Key* i *Text*
  - Za vrijednost u padajućoj listi korišteno svojstvo *Key*
  - Za prikaz teksta u padajućoj listi korišteno svojstvo *Text* iz *LookupData*
  - Odabrani element se veže na konkretno svojstvo iz Dokumenta (npr. na *IdPartnera*)

```
<asp:DropDownList ID="ddlPartneri" runat="server"  
    SelectedValue='<%# Bind("IdPartnera") %>'  
    DataSourceID="PartnerDataSource"  
    DataTextField="Text"  
    DataValueField="Key" />
```

# Odabir vrijednosti stranog ključa padajućom listom (2)

## ❑ Primjer: FirmaWebForms – DokumentStavka.aspx

- Moguće kombinirati fiksne vrijednosti i vrijednosti iz izvora
- Svojstvo *AppendDataBoundItems* se postavlja na *true*
- Fiksni se navode kao elementi tipa *ListItem* s atributima *Key* i *Value*

```
<asp:DropDownList ID="ddlPrethodniDokument" runat="server"
    AppendDataBoundItems="True"
    SelectedValue='<%# Bind("IdPrethDokumenta") %>'
    DataSourceID="PrethodniDokumentDataSource"
    DataTextField="Text"
    DataValueField="Key">
    <asp:ListItem Value="">Nema prethodnog dokumenta</asp:ListItem>
</asp:DropDownList>
```

# Predložak za dodavanje novog dokumenta

## ❏ Primjer: FirmaWebForms – DokumentStavka.aspx

- *FormView* se prebacuje u *Insert* način rada klikom na gumb vezan uz naredbu *New* i vraća se u “normalni” način prikaza klikom na gumb koji je vezan uz naredbu *Cancel*
- Dizajnira se unutar oznake *InsertTemplate* unutar *FormView*a
- Gotovo identičan predlošku za ažuriranje
- Na dnu predloška dodan gumb za potvrdu unosa (naredba *Insert*) i gumb za odustajanje (naredba *Cancel*)
  - Za sve kontrole za unos dodati validacijske kontrole i na njima postaviti svojstvo *ValidationGroup* na istu vrijednost kao i kod gumba za spremanje promjena

```
<asp:TextBox ID="tbPostoPorez" runat="server"
    Text='<%# Bind("PostoPorez") %>' />
<asp:RequiredFieldValidator ID="RequiredFieldValidatorPorez"
runat="server" ControlToValidate="tbPostoPorez" ErrorMessage="*"
ValidationGroup="Dokument"/>CausesValidation="True"
CommandName="Insert" Text="Spremi" ValidationGroup="Dokument" />

<asp:Button ID="InsertCancelButton" runat="server"
CausesValidation="False" CommandName="Cancel" Text="Odustani" />
```

# Specifičnosti kod predloška za dodavanje

## ❑ Primjer: Firma.WebForms – DokumentStavka.aspx.cs

- Obrađuje se događaj *ItemCreated* na *FormViewu*
  - nakon što se kontrole iz predloška stvore, a prije povezivanja podataka
  - događaj nije vezan samo za dodavanje te se provjerava trenutni način prikaza
- Svaka kontrola iz predloška se može dohvatiti postupkom *FindControl* npr. *FormViewDokument.FindControl("NazivKontrole")*
- GridView sa stavkama se treba sakriti ovisno o načinu prikaza ili ako nema podataka
- Ako ne postoji niti jedan dokument, prebaciti se odmah u način za dodavanje
- Prije početka dodavanja postavljaju se inicijalne vrijednosti za datum, br.dokumenta...

```
protected void FormViewDokument_ItemCreated(object sender, ...) {  
    FormView form = (FormView)sender;  
    if (form.CurrentMode == FormViewMode.Insert) {  
        TextBox tbDatum =  
            (TextBox) FormViewDokument.FindControl ("tbDatDokumenta");  
        ...  
        GridViewStavke.Visible = false;  
    }  
    else if (form.CurrentMode == FormViewMode.ReadOnly &&  
            form.DataItemCount == 0) {  
        form.ChangeMode(FormViewMode.Insert);  
        GridViewStavke.Visible = false;  
    }  
}
```

# Dodavanje novog dokumenta

## ❑ Primjer: FirmaWebForms – DokumentStavka.aspx.cs

- Gumb s naredbom Insert (unutar predloška *InsertTemplate* s prethodnog slajda) aktivira postupak na izvoru podataka (pridružen *InsertMethod*)

```
<asp:ObjectDataSource ID="DokumentDataSource" ...  
TypeName="DokumentController" InsertMethod="InsertDokument"
```

## ■ Primjer: FirmaWebForms – DataControllers\DokumentController.cs

- Vrijednosti povezanih podataka šalju se pridruženom postupku

```
public class DokumentController {  
    public void InsertDokument(string VrDokumenta,  
        int BrDokumenta, string DatDokumenta, int IdPartnera,  
        string PostoPorez, int? IdPrethDokumenta) {  
        Dokument dokument = new Dokument();  
        dokument.VrDokumenta = VrDokumenta;  
        dokument.BrDokumenta = BrDokumenta;  
        dokument.DatDokumenta = DateTime.ParseExact(  
            DatDokumenta, "dd.MM.yyyy", null);  
        ...  
        DokumentBllProvider bll = new DokumentBllProvider();  
        bll.Save(dokument);  
    }  
}
```

# Brisanje dokumenta

## ❑ Primjer: FirmaWebForms – DokumentStavka.aspx

- Unutar predloška za prikaz (*ItemTemplate*) definiran gumb s akcijom brisanja

```
<asp:Button ID="DeleteButton" ... CausesValidation="False"
           CommandName="Delete" Text="Obriši" .../>
```

- Klikom na gumb poziva se postupak pridružen brisanju kojem se proslijeđuju vrijednosti primarnog ključa (*IdDokumenta* u ovom primjeru)

```
<asp:FormView ID="FormViewDokument" DataKeyNames="IdDokumenta"
```

```
<asp:ObjectDataSource ID="DokumentDataSource" ...
TypeName="DokumentController" DeleteMethod="ObrisiDokument"
```

- Primjer:  FirmaWebForms – DataControllers\DokumentController.cs

```
public class DokumentController {
    public void ObrisiDokument(int IdDokumenta) {
        var bll = new DokumentBllProvider();
        var dokument = bll.Fetch(IdDokumenta);
        dokument.Delete();
        bll.Save(dokument);
    }
}
```

- Brisanje stavke izvedeno na sličan način

# Specifičnosti kod brisanja

## ❑ Primjer: FirmaWebForms – DokumentStavka.aspx.cs

- Obrađuje se događaj *ItemDeleted* na *FormViewu*
- Potrebno ponovno aktivirati povezivanje za *FormView* kako bi se automatski dogodilo povezivanje vezanih podataka (stavke dokumenta)
- Ponašanje forme pri brisanju: broj stranice ostaje isti
  - iznimka predzadnja stranica – broj stranice se umanjuje za 1 (\*nije dokumentirano, praksa pokazala)
- Što ako brišemo podatak s posljednje stranice? Potrebno smanjiti broj stranice.

```
protected void FormViewDokument_ItemDeleted
    (object sender, FormViewDeletedEventArgs e)
{
    FormView form = (FormView)sender;
    if (form.PageIndex == form.PageCount - 1)
    {
        --form.PageIndex;
    }
    FormViewDokument.DataBind();
}
```



# Prikaz stavki nekog dokumenta (1)

## □ Primjer: FirmaWebForms – DokumentStavka.aspx

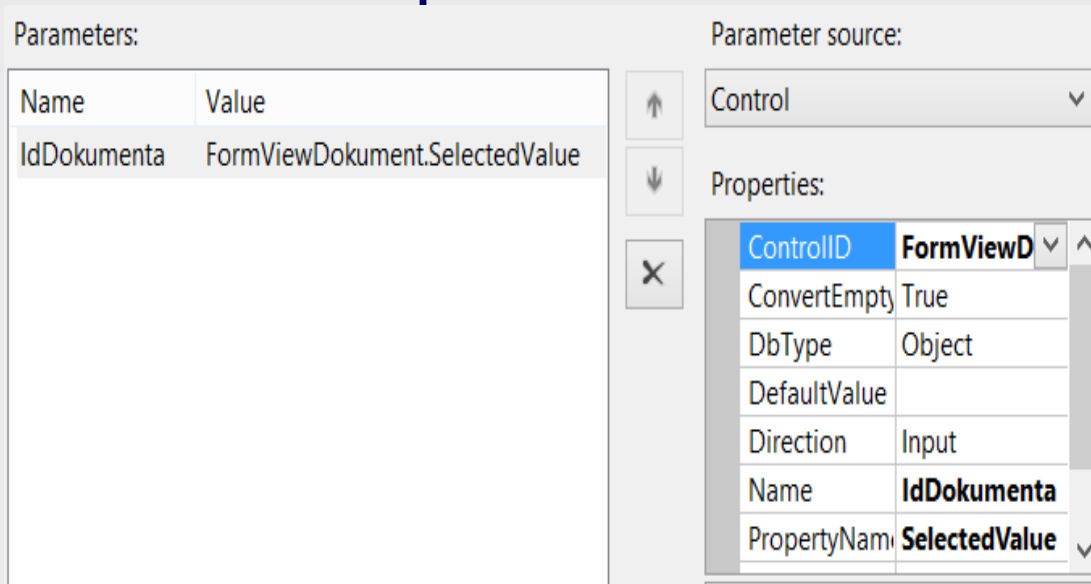
- Stavke se prikazuju unutar *GridViewa*
- Izvor je tipa *ObjectDataSource*
- Razred `DataController\DokumentController.cs` vrši dohvat koristeći `Firma.BLL.dll` iz primjera `Firma.Win`
- Dohvat parametriziran trenutno prikazanim dokumentom (vidi sljedeći slajd)

Properties	
<b>StavkeDataSource</b> System.Web.UI.WebControls.ObjectDataSource	
DeleteParameters	(Collection)
EnableCaching	False
EnablePaging	False
EnableViewState	True
FilterExpression	
FilterParameters	(Collection)
InsertMethod	
InsertParameters	(Collection)
MaximumRowsParameterName	maximumRows
OldValuesParameterName	{0}
SelectCountMethod	
SelectMethod	<b>DohvatiStavkeDokumentom</b>
SelectParameters	(Collection)
SortParameterName	
SqlCacheDependency	
StartRowIndexParameterName	startRowIndex
TypeName	<b>DokumentController</b>
UpdateMethod	<b>AzurirajStavku</b>

# Prikaz stavki nekog dokumenta (2)

## □ Primjer: FirmaWebForms – DokumentStavka.aspx

- Prikaz ograničen samo na stavke trenutnog dokumenta
- Uzima se trenutni dokument u *FormViewu*
- Svojstvo *SelectParameters*
  - Postavlja se kroz dizajn ili ručnim dopisivanjem unutar kontrole *ObjectDataSource*
- Postupak za dohvat mora primiti navedene parametre



The screenshot shows the Visual Studio IDE with two windows open for an *ObjectDataSource* control:

- Parameters:** A table with two columns: *Name* and *Value*. It contains one parameter:

Name	Value
IdDokumenta	FormViewDokument.SelectedValue
- Parameter source:** A dropdown menu set to *Control*.
- Properties:** A list of properties for the *FormViewDokument* control:

Property	Value
ControlID	FormViewDokument
ConvertEmpty	True
DbType	Object
DefaultValue	
Direction	Input
Name	IdDokumenta
PropertyName	SelectedValue

```
<asp:ObjectDataSource ID="StavkeDataSource" runat="server"
    TypeName="DokumentController"
    SelectMethod="DohvatiStavkeDokumenta"
    DeleteMethod="ObrisiStavku" UpdateMethod="AzurirajStavku" >

    <SelectParameters>
        <asp:ControlParameter ControlID="FormViewDokument"
            Name="IdDokumenta" PropertyName="SelectedValue" />
    </SelectParameters>
</asp:ObjectDataSource>
```

# Oblikovanje pojedinog stupca u mreži sa stavkama

## ❑ Primjer: FirmaWebForms – DokumentStavka.aspx

- Svako polje (stupac) izvedeno kao `TemplateField`
  - Predložak za prikaz (*ItemTemplate*), predložak za ažuriranje (*EditItemTemplate*) i predložak za podnožje (*FooterTemplate*) kojim se omogućava dodavanje novog podatka
  - Ista pravila za povezivanje podataka kao i kod *FormView* (*Eval*, *Bind*)
- Predložak za situaciju u kojoj nema podataka u mreži
  - *EmptyDataTemplate* – dizajnira se za *GridView*, ne za pojedini stupac
- Validacijske kontrole s drugačijom vrijednošću svojstva *ValidationGroup* kako ne bi bili u koliziji s validacijom kod ažuriranje dokumenta
- U konkretnom primjeru obrađuju se sljedeći događaji
  - *RowDeleted* i *RowUpdated* – potreba da se ažurira cijena dokumenta
  - *RowUpdating* – potreba da se dohvati cijena odabranog artikla i prekopira u konkretnu stavku
  - *RowCommand* – potreba da se obrade vlastite naredbe koje nisu iz standardnog skupa naredbi (*Edit*, *Update*, *Cancel*, *Delete*, *New*)
    - Dvije vlastite naredbe (*Add*, *AddForEmpty*) za dodavanje novog podatka iz podnožja te iz predloška koji se prikazuje kad nema podataka u mreži

# Dodavanje nove stavke (obrađivanje događaja *RowCommand*)

## ❏ Primjer: FirmaWebForms – DokumentStavka.aspx.cs

- Ako se dodavanje odvija iz podnožja mreže
  - Dohvat kontrole postupkom *FindControl* na retku podnožja  
`GridViewStavke.FooterRow.FindControl("ddlSifArtikla");`
- Ako se stavka dodaje iz predloška za praznu mrežu
  - Specifičan način dohvata pojedine kontrole:  
`GridViewStavke.Controls[0].Controls[0].  
FindControl("ddlSifArtikla");`
- Stvoriti stavku i dodati je u konkretni dokument te snimiti promjene

```
protected void GridViewStavke_RowCommand...  
    int IdDokumenta = (int)FormViewDokument.SelectedValue;  
    var bll = new DokumentBllProvider();  
    Dokument dokument = bll.Fetch(IdDokumenta);  
    ...  
    Stavka stavka = new Stavka(){ SifArtikla = artikl.SifArtikla, ... };  
    ...  
    dokument.Edit();  
    dokument.Stavke.Add(stavka);  
    bll.Save(dokument);
```

- Nakon dodavanja pozvati *DataBind* na *GridViewu*, a zatim i na *FormViewu*

```
GridViewStavke.DataBind(); FormViewDokument.DataBind();
```

# Reference i dodaci

- RPPP10-dodatak.pdf
- ASP.NET Web Forms
  - <http://www.asp.net/web-forms>
- w3schools.com
  - <http://www.w3schools.com/>