

ASP.NET Web Formları ile Çalışmak

Cengiz HAN

cengiz@cengizhan.com

www.cengizhan.com



1

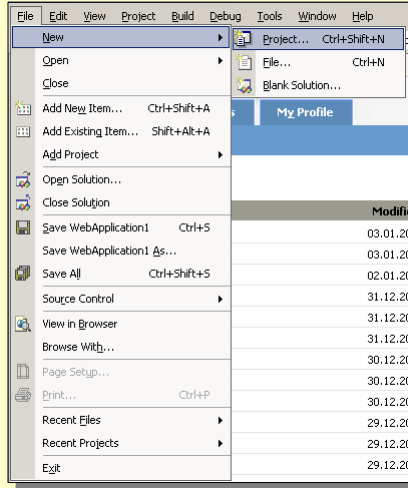
Microsoft ASP.NET Web Formları ile Çalışmak

- Konu 1: Web Formları Oluşturmak
- Konu 2: Sunucu Kontrollerini Kullanmak
- Konu 3: Arka Planda Kod İçeren(Code-Behind) Sayfalar
- Konu 4: Web Sunucu Kontrollerine Olay Prosedürleri Ekleme
- Konu 5: Sayfa Olaylarını Kullanmak

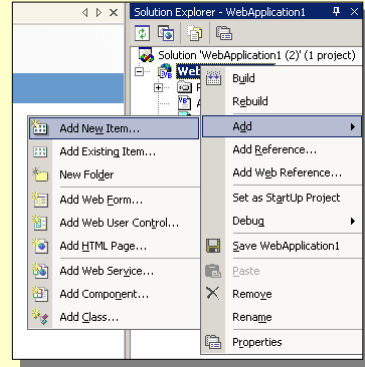
2

Visual Studio.NET ile Web Form Oluşturmak

Yeni Web Uygulaması Oluşturmak

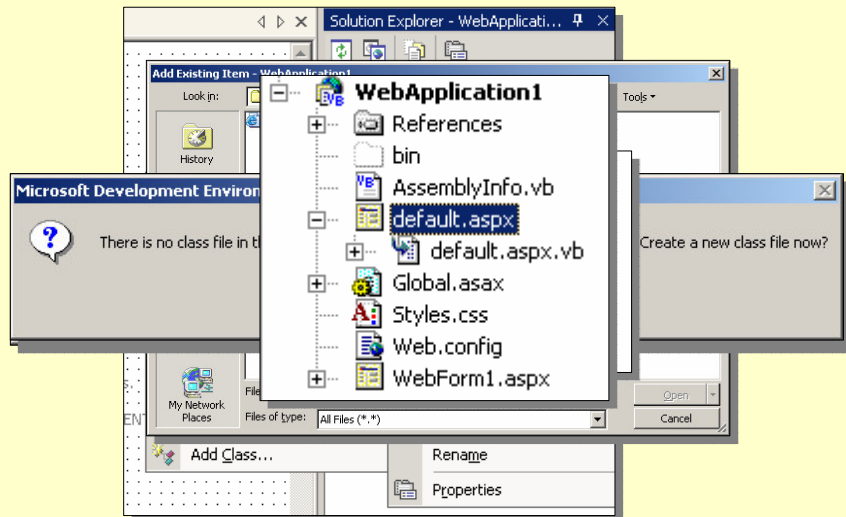


Bir Projeye Web Form Ekleme



3

HTML Sayfasını ASP.NET Sayfasına Çevirmek



4

Konu 2: Sunucu Kontrollerini Kullanmak

Sunucu Kontrolleri Nedir?

- ASP.NET Sunucu Kontrolleri sunucu tarafında çalışan bileşenlerdir.
 - button, textbox, listbox v.b. basit çıktı üreten kontroller
 - calendar, datagrid gibi gelişmiş çıktı üreten kontroller

```
<INPUT id="Text1" type="text" name="Text1" runat="server">
```

```
<asp:TextBox id="TextBox1" runat="server"></asp:TextBox>
```

- Sunucu Kontrolleri runat="server" özelliğine sahiptirler ve mutlaka sunucu tarafında çalışan bir form içerisinde yer almalıdır.

```
<form id="Form1" method="post" runat="server"></form>
```

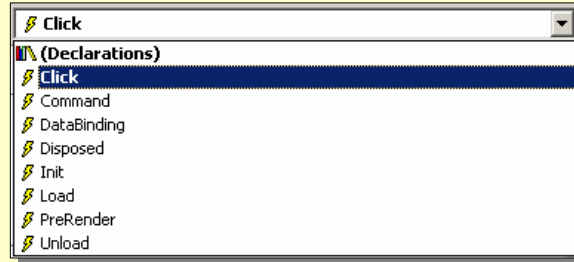
5

Olaya Dayalı (Event Driven) Programlama

- Web Form'unda bulunan bir düğmeye tıklanması, bir listede bulunan seçili elemanın değişmesi birer olaydır(event).
- Her sunucu kontrolünün kendine has olayları vardır.
- Olayların oluşması anında çalışacak prosedürler tanımlanır.

Bir Button Web Sunucu Kontrolünün Olayları

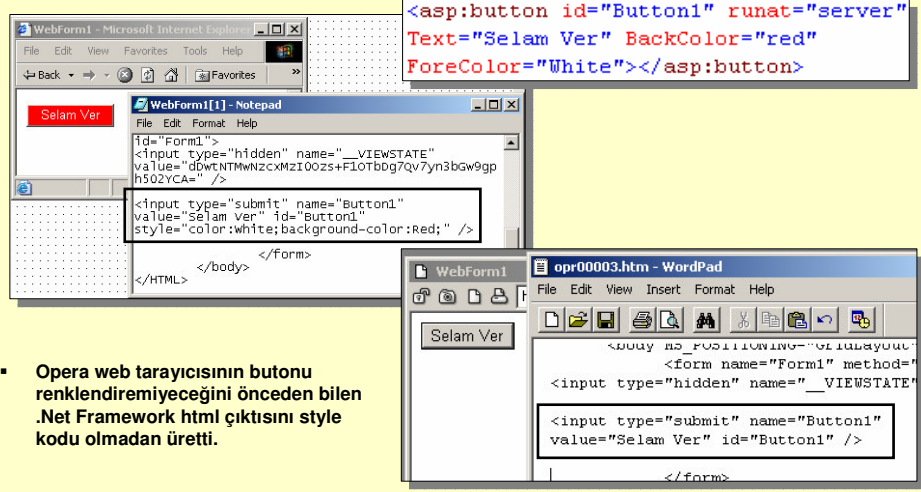
Visual Studio.NET Kod Editör'ü listeler



6

Genel Nesne Modeli

- ASP.NET Sunucu Kontrolleri genel bir nesne modeline sahiptir.
- Bazı özellikleri tüm sunucu kontrolleri barındırır.



- Opera web tarayıcısının butonu renklendiremeyeceğini önceden bilen .Net Framework html çıktısını style kodu olmadan üretti.

Sunucu Kontrol Tipleri

- Html Sunucu Kontrolleri (Html Server Controls)
 - Sunucu tarafından çalışan kod ile bir html ögesine erişmek için `runat="server"` ifadesinin html etiketine eklenmesi gerekir.
 - `System.Web.UI.HtmlControls.HtmlInputText`
 - `Text1.Value = "Merhaba!!!"`
 - Html sözdizimi ile örtüşür.

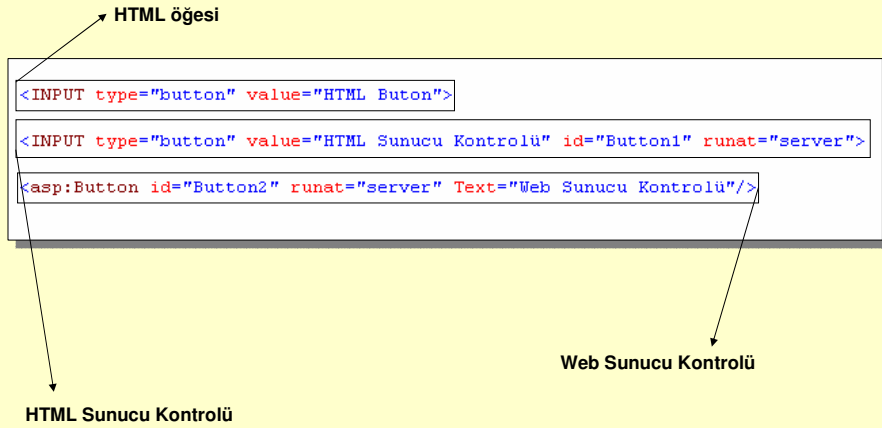
```
<INPUT id="Text1" type="text" name="Text1" runat="server">
```

- Web Sunucu Kontrolleri (Web Server Controls)
 - `System.Web.UI.WebControls.TextBox`
 - `TextBox1.Text = "Merhaba!!!"`
 - Kutuda yazan metin Text özelliği ile belirlenir.

```
<asp:TextBox id="TextBox1" runat="server"></asp:TextBox>
```

- Web Kontrolleri aşağıdaki gibi bazı kategorilere ayrılabilir.
 - Gerçek (Intrinsic) Kontroller (Buton, ListBox, TextBox gibi)
 - Geçerlilik (Validation) Kontrolleri (RequiredFieldValidator, RegularExpressionValidator gibi)
 - Zengin Kontroller (AdRotator, Calendar gibi)
 - Veri Bağlantılı Kontroller (DataList, Repeater gibi...)

Sunucu Kontrolleri Uygulamaları



9

ViewState

- Temelde yatan mimariye göre web sitelerinin başlıca problemlerden birisi sayfanın istemci(client) ve sunucu(server) arasında gidip gelmesi aşamasında sayfadaki kontrollerin durumlarının nasıl saklanacağıdır.
- ASP.NET bu durum için tamamen otomatik bir yönetim sağlar. Sayfamız sunucuya postalanıp tekrar geri geldiğinde form değerleri saklanır.

```
<form name="Form1" method="post"
action="webForm1.aspx" id="Form1">

<input type="hidden" name="__VIEWSTATE"
value="d0wtMTA2MzcyMDk3NTt0PHA8bdx40z47bdwgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
gICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
4+0zS+0z44D5FZdAm3/A7n66LLMNv1SmPgBQ==" />

<INPUT type="button" value="HTML Buton">

<input name="Button1" id="Button1" type="button"
value="HTML Sunucu Kontrolü" />
```

```
ViewState("sehir") = "İzmir"
Label1.Text = ViewState("sehir")
```

```
ViewState["sehir"]="İzmir";
Label1.Text=(string)ViewState["sehir"];
```

10

ViewState Özelliğini Kapatmak

- Herhangi bir değişiklik yapılmadığı takdirde web formdaki kontrollerin durum bilgileri saklanır.
- Çok fazla kontrol içeren sayfalarda viewstate hidden(gizli) değerinin boyutu sayfanın boyutunu artırıp sayfanın yavaşlamasına sebep olabilir.

```
<%@ Page EnableViewState="False"
```

```
<asp:ListBox id="lstUlke" runat="server"  
EnableViewState="False"></asp:ListBox>
```

11

HTML Sunucu Kontrolleri

- Geleneksel Html öğeleri sunucu tarafından çalışan kod tarafından erişilemez.
- Bu öğeleri sunucu kontrollerine çevirdiğiniz zaman sunucu tarafında çalışan kodlar tarafından erişilebilir ve olayları(event) olan HTML Sunucu Kontrolleri olurlar.

```
<INPUT id="Text1" type="text" name="Text1" runat="server">
```

- System.Web.UI.HtmlControls namespace' i içinde yer alan sınıflardır.
 - System.Web.UI.HtmlControls.HtmlInputText
- Html söz dizimi ile örtüşür.
 - Text1.Value = "Merhaba!!!"

12

Html Öğesini Html Sunucu Kontrolüne Çevirmek

- Html Öğesi :

```
<input type="text" id="txtAdi">
```

- Html öğesine **runat="server"** özelliğini ekleyerek yazı kutusunu bir HTML Sunucu Kontrolüne çevirebiliriz.

```
<input type="text" id="txtAdi" runat="server"/>
```

13

Web Sunucu Kontrolleri

- Web Sunucu Kontrolleri özellikle ASP.NET için hazırlanmış olan sunucu kontrolleridir.
- Web Sunucu Kontrolleri genel bir nesne modeline sahiptir. Sayfaya eklemek için **<asp:KontrolTipi ...>** gibi bir tanımlama kullanılır.
- Programatik erişim için kullanılacak olan **id** özelliğine sahiptir.
- System.Web.UI.WebControls namespace

```
<asp:TextBox id="TextBox1" runat="server"></asp:TextBox>
```

14

Gerçek (Intrinsic) Web Sunucu Kontrolleri

- Bu sunucu kontrolleri basit html öğelerine karşılık gelmektedir.
- Gerçek (Intrinsic) Web Sunucu Kontrolleri ve Html Karşılıkları

Web Sunucu Kontrolü	Html karşılığı
<asp:button>	<input type=submit>
<asp:checkbox>	<input type=checkbox>
<asp:hyperlink>	
<asp:image>	
<asp:imagebutton>	<input type=image>
<asp:linkButton>	Yok
<asp:label>	
<asp:listbox>	<select size="5"></select>
<asp:panel>	<div></div>
<asp:radiobutton>	<input type=radiobutton>
<asp:table>	<table> </table>
<asp:textbox>	<input type=text>

15

Geçerlilik(Validation) Kontrolleri

- CompareValidator
- CustomValidator
- RangeValidator
- RegularExpressionValidator
- RequiredFieldValidator
- ValidationSummary

16

Zengin Kontroller (Rich Controls)

- AdRotator

- Sayfada görüntülenecek resimleri

```
<?xml version="1.0" encoding="utf-8" ?>
<Advertisements>
- <Ad>
  <ImageUrl>images/ba.gif</ImageUrl>
  <NavigateUrl>http://www.yazgelistir.com</NavigateUrl>
  <AlternateText>Yazgelistir.com</AlternateText>
  <Keyword>dotnet</Keyword>
  <Impressions>80</Impressions>
</Ad>
- <Ad>
  <ImageUrl>images/aspnet.gif</ImageUrl>
  <NavigateUrl>http://www.asp.net</NavigateUrl>
  <AlternateText>ASP.NET</AlternateText>
  <Keyword>dotnet</Keyword>
  <Impressions>80</Impressions>
</Ad>
</Advertisements>
```

```
<asp:AdRotator id="AdRotator1"
runat="server"
AdvertisementFile="ads.xml"
KeywordFilter="dotnet">
</asp:AdRotator>
```

```
<a id="AdRotator1" href="http://www.yazgelistir.com"
target="_top"></a>
```

- Calendar

Ocak 2004							
Pzt	Sal	Çar	Per	Cum	Cmt	Paz	
29	30	31	1	2	3	4	
5	6	7	8	9	10	11	
12	13	14	15	16	17	18	
19	20	21	22	23	24	25	
26	27	28	29	30	31	1	
2	3	4	5	6	7	8	

17

Veri Bağlantılı Kontroller

- CheckBoxList
- Repeater
- Datalist
- DataGrid
- DropDownList
- ListBox
- RadioButtonList

18

Uygun Sunucu Kontrolünü Seçmek

Ne zaman HTML Sunucu Kontrolleri?

- HTML benzeri bir nesne modeli kullanmak istendiği zaman,
- Halen çalışmakta olan HTML sayfalarını ASP.NET sayfalarına **hızlıca** çevirmek istendiğinde,
- Kontrol hem istemci(client) tarafında bir script kodu çalıştıracak, hemde sunucu tarafı kod çalıştıracak ise,

Ne Zaman Web Sunucu Kontrolleri?

- Visual Basic benzeri programlama yapmak istendiği durumlarda,
- Çeşitli web tarayıcıları tarafından görüntülenecek sayfalar hazırlanacağı durumlarda,
- Geleneksel HTML öğeleriyle yapılamayacak işlerin olduğu özel durumlarda tercih edilir.
 - Calendar, DataGrid

19

Konu 3: Arka Planda Kod İçeren(Code-Behind) Sayfalar

▪ Sayfaya Kod Eklenmesi

- **ASP.NET sayfasına üç şekilde kod ekleyebiliriz.**
 - **Karışık Kod** : Program kodu ve web içeriği(HTML) aynı dosyada yer alır.
 - **Satır İçi (Inline) Kod** : Program kod ve web içeriği aynı dosyada yer alır. Fakat program kodları SCRIPT bölümü içerisinde html içeriğinden ayrı olarak yazılır.
 - **Arka Planda Kod(Code-Behind)** : Program kodu ve HTML web içeriği ayrı dosyalarda yazılır.

20

Satır İçi (Inline) Kod Yazmak

- Visual Studio .NET ile kullanılan varsayılan metod code-behind yöntemidir.
- Satır içi(inline) kod yazımı özellikle ASP 3.0 dan ASP.NET geçiş sürecinde başvurulan bir yöntemdir.

```
<%@ Page Language="VB" %>
<script runat="server">
    Sub Button1_Click(sender As Object, e As EventArgs)
        Button1.Text="Merhaba"
    End Sub
</script>
<html>
<head>
</head>
<body>
    <form runat="server">
        <asp:Button id="Button1" onclick="Button1_Click"
            runat="server" Text="Button"></asp:Button>
    </form>
</body>
</html>
```

21

Code-Behind (Arka Planda Kod İçeren) Sayfalar

- Visual Studio .NET ASP.NET sayfalarına kod eklemek için varsayılan olarak code-behind yöntemini kullanır.
- Code-Behind yöntemini kullandığınızda program kodu ile görsel elemanları oluşturan html kodu ayrı dosyalarda saklanır.
- **Code-Behind Sayfalarının İsimlendirilmesi**
 - Bir web uygulamasında her web sayfası kendi code-behind sayfasına sahiptir.
 - Webform1.aspx sayfasının code-behind sayfası Webform1.aspx.vb (Webform1.aspx.cs)

22

Code-Behind Sayfaları Nasıl Çalışır?

- Code-Behind sayfalarının çalışması için her .aspx sayfası bir Code-Behind sayfası ile ilişkilendirilmiş olmalıdır.
- Ayrıca sayfanın çalışması için code-behind dosyaları derlenmiş (compiled) olmalıdır.
- Code-Behind dosyaları tek bir Assembly'e derlenirler.

23

Code-Behind Dosyası Bildirimi

- .aspx dosyası bir code-behind dosyası ile ilişkilendirilmelidir.
- Visual Studio .NET aşağıdaki özellikleri @Page bildirimine ekler.
 - **Codebehind**
 - Bu özellik Visual Studio .NET Web Forms Designer tarafından codebehind dosyasını tanımlamak için kullanılır. Bu özellik çalışma anında kullanılmaz.
 - **Inherits**
 - Miras alınacak sınıfın adı
 - Bu sınıf System.Web.UI.Page sınıfını miras almış olmalıdır.

```
<%@ Page Language="vb"  
Codebehind="WebForm1.aspx.vb"  
Inherits="MerhabaDunya.WebForm1"%>
```

24

JIT(Just In Time) Derleme

- Bu yöntem ile derleme işlemi gerektiği anda yapılır, yani bir .aspx dosyasına istek geldiğinde anda derleme yapılır.
- İlk istekten sonra takip eden istekler daha önce derlenmiş olan dosyaları kullanır.
- Bu durumda sayfanın ilk isteği biraz yavaş olur fakat takip eden istekler derlenmiş dosyayı kullandığı için daha hızlı olur.
- Eğer sayfanız için JIT derleme yöntemini kullanmak istiyorsanız @Page etiketinde Src özelliğini kullanmalısınız.
- JIT derleme sağladığı bir kolaylık ise tüm siteyi tekrar derlemeye gerek kalmadan program kodu düzenlemesi yapılabilmesidir.

```
<%@ Page Language="vb" Src="WebForm1.aspx.vb" Inherits="WebForm1"%>
```

25

Konu 4: Web Sunucu Kontrollerine Olay Prosedürleri Ekleme

- Olay Prosedürleri Oluşturmak
 - Dinamik ve etkileşimli web sayfaları kullanıcı hareketlerine tepki verir.
 - Olay prosedürleri bu durumları algılayıp işlemek için kullanılır.
 - Kullanıcı web form ile etkileşime geçtiği zaman olaylar oluşur.
- İstemci Taraflı
- Sunucu Taraflı

26

İstemci Taraflı Olay Prosedürleri

- Web sayfasını görüntüleyen kullanıcıların tarayıcılarında çalışır.
- Html öğeleri ile kullanılabilir.
- Sunucu kaynaklarına erişemez.
- İstemci taraflı olaylar sunucu-istemci arasında gidiş-geliş işlemine gerek olmayan hızla istediğinizi yapmak istediğiniz durumlarda kullanışlıdır.
- Tabiki istemci tarafında yapılabilecek işlemler sunucu kaynaklarına erişemez.
- Bir yazı kutusunda yazan bilginin sunucuya postalanmadan önce boş olmadığının kontrolü gibi işlemler.

```
<script language="javascript">  
    //istemci taraflı kod  
</script>
```

27

Sunucu Taraflı Olay Prosedürleri

- İstemci taraflı olay prosedürlerinden farklı olarak bilginin web sunucuya gönderilmesini gerektirir.
- Sunucu taraflı olay prosedürleri daha fazla zaman kullanılmasına sebep olsa da çok daha fazla güçlü ve işlevseldir.
- Sunucu kaynaklarına erişerek işlem yapabilmeyi sağlar.
- Sunucu taraflı olay prosedürleri web sunucuda bulunan derlenmiş program kodundan oluşur.

28

Olay Prosedürleri Oluşturmak

- İlk olarak sayfayı olayı oluşturacak bir sunucu kontrolü eklenir.
- VS.NET içerisinde bir sunucu kontrolünün üzerine çift tıklamak...
- Handles anahtar kelimesi
 - AutoEventWireup özelliği
- EventArgs


```
<form id="Form1" method="post" runat="server">  
  <asp:Button id="btnKaydet" runat="server" Text="Button"></asp:Button>  
</form>
```

```
Protected WithEvents btnKaydet As System.Web.UI.WebControls.Button  
Private Sub Page_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load  
    'Put user code to initialize the page here  
End Sub  
Private Sub btnKaydet_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnKaydet.Click  
End Sub
```

29

Olay Prosedürlerinden Kontrollere Erişim ve Sunucu Kontorülüne Yazmak

```
Private Sub btnKaydet_Click(ByVal sender As System.Object  
    Dim strMesaj As String = "Merhaba " & txtIsim.Text  
    btnKaydet.Text = strMesaj  
End Sub
```



The screenshot shows a web form with a text input field containing the text 'Ali'. To the right of the input field is a button that has been updated with the text 'Merhaba Ali', demonstrating the result of the server-side code.

30

Konu 5: Sayfa Olaylarını Kullanmak

- Page.Init
- Page.Load
- Kontrol Olayları
- Page.Unload
- **Örnek uygulama;**
Birkaç textbox kontrolü bulunan bir web formunu görüntüleyen bir kullanıcı textbox kontrollerine bilgileri girdikten sonra Gönder buton kontrolüne tıklar.
- *Tıklama anına kadar sayfa sunucuya geri gönderilmez ve TextBox kontrollerinin change olayları tetiklenmez, tepkisiz bir halde bekler.*
- *Butona tıklanıldığı zaman sayfa sunucuya geri gönderilir. Sayfanın Load olayı çalıştıktan sonra, Change olayları sonra Click olayı çalışır.*
- ASP.NET formları sunucu tarafı işlemlerin yapılabilmesi için bilgiyi aynı .aspx dosyasına tekrar postalayacak şekilde tasarlanmıştır. Bu işlem **PostBack** (Geriye Postalama) olarak isimlendirilmiştir.

31

AutoPostBack

- Varsayılan olarak sadece Click (tıklanma) olayları formları geriye postalar, Ancak bazı kontroller **AutoPostBack** özellikleri true yapıldığı takdirde geriye postalama yapıp sunucu olay prosedürlerinin çalışmasını sağlayabilmektedir.

```
Private Sub ListBox1_SelectedIndexChanged(ByVal sender As Object, ByVal e As EventArgs)
    TextBox1.Text = ListBox1.SelectedValue
End Sub
```

```
<asp:DropDownList id="ListBox1" runat="server" AutoPostBack="True">
    <asp:ListItem>Bornova</asp:ListItem>
    <asp:ListItem>Karşıyaka</asp:ListItem>
</asp:DropDownList><BR>
<asp:TextBox id="TextBox1" runat="server"></asp:TextBox>
```

32

Page.IsPostBack Kontrolü

- Page.Load ilk istekte de, postback isteklerinde de tetiklenir.
 - Bu olayda yazılan tüm program kodları her seferinde çalışır.
 - Yazılan kodların sadece ilk sefer de çalışması istenen durumlarda Page.IsPostBack özelliği kontrol edilir.
 - Bu özellik eğer sayfa **ilk defa çağrılıyorsa false, postback isteğinde çağrılıyorsa true** boolean değerini geri döner.

```
Private Sub Page_Load(ByVal s As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    If Not Page.IsPostBack Then
        'sadece sayfaya gelen ilk istekte çalışır
        mesleklerilistele()
    Else
        'Buradaki kod sadece postback
        'durumlarında çalışır.
    End If
    'buradaki kod sayfaya gelen her istekte çalışır.
End Sub
```

33

İki Kontrolü Birbirine Bağlamak

- Bir kontrolü başka bir kontrolün içeriğine bağlanabilmektedir.
- <control>.DataBind metodu bir kontrolü ve onun tüm alt kontrollerinin bağlantısını sağlar.
- Yani her kontrolün bağlantılarının işlemesi için databind methodu çağrılmalıdır.
- Çok fazla kontrol barındıran bir Web Form'larda her kontrol için bu metodu çağırmak yerine Page.DataBind metodu çağırılabilir.
- Databind metodu çağrılan kontrolün tüm alt kontrollerini de etkiler.

```
<asp:DropDownList id="lstMeslek" AutoPostBack="true" runat="server">
    <asp:ListItem>Yazılım Geliştirici</asp:ListItem>
    <asp:ListItem>Grafik Tasarımcısı</asp:ListItem>
    <asp:ListItem>Sistem Destek Uzmanı</asp:ListItem>
</asp:DropDownList>
<br>Seçili Meslek:
<asp:Label id="lblMeslek" Text="<%# lstMeslek.SelectedItem.Text %>"
runat="server" />
```

```
Private Sub Page_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    lblMeslek.DataBind() 'Page.DataBind()
End Sub
```

34

GÖZDEN GEÇİRME

1. Web Form'larında Code-behind sayfaları kullanmanın avantajları nelerdir?
2. Bir sunucu kontrolü olayı ile ilişkilendirilmiş olan bir prosedürün yapısı nasıl olmalıdır?
3. Olay prosedürlerinin iki parametresi nedir?
4. Bir .aspx sayfası ile code-behind dosyası nasıl ilişkilendirilir?
5. ASP.NET sayfalarına program kodu ekleyebileceğiniz 3 yöntemi listeleyiniz?
6. Code-behind dosyalarını neden JIT (just-in-time / gerektiği zaman derleme) derleme yerine önceden derlenmiş (assembly-dll) olarak kullanmak tavsiye edilir?
7. Bir Web Form ne zaman postback olur?
8. Page.Load olay prosedüründe sayfanın ilk defa mı çağrıldığını yoksa postback mi olduğunu nasıl belirleyebilirsiniz?

35

UYGULAMA

1. Html web sayfasını ASP.NET Web Form sayfasına çevirilmesi.
2. Sunucu olay prosedürlerini ile çalışmak. Olayların sırası.
3. İki kontrolün birbirine bağlanması.

36