

T.C.
MİLLÎ EĞİTİM BAKANLIĞI



MEGEP

(MESLEKİ EĞİTİM VE ÖĞRETİM SİSTEMİNİN GÜÇLENDİRİLMESİ
PROJESİ)

BİLİŞİM TEKNOLOJİLERİ

ETKİLEŞİMLİ WEB UYGULAMALARI - 3

ANKARA 2008

Milli Eğitim Bakanlığı tarafından geliştirilen modüller;

- Talim ve Terbiye Kurulu Başkanlığının 02.06.2006 tarih ve 269 sayılı Kararı ile onaylanan, Mesleki ve Teknik Eğitim Okul ve Kurumlarında kademeli olarak yaygınlaştırılan 42 alan ve 192 dala ait çerçeve öğretim programlarında amaçlanan mesleki yeterlikleri kazandırmaya yönelik geliştirilmiş öğretim materyalleridir (Ders Notlarıdır).
- Modüller, bireylere mesleki yeterlik kazandırmak ve bireysel öğrenmeye rehberlik etmek amacıyla öğrenme materyali olarak hazırlanmış, denenmek ve geliştirilmek üzere Mesleki ve Teknik Eğitim Okul ve Kurumlarında uygulanmaya başlanmıştır.
- Modüller teknolojik gelişmelere paralel olarak, amaçlanan yeterliği kazandırmak koşulu ile eğitim öğretim sırasında geliştirilebilir ve yapılması önerilen değişiklikler Bakanlıkta ilgili birime bildirilir.
- Örgün ve yaygın eğitim kurumları, işletmeler ve kendi kendine mesleki yeterlik kazanmak isteyen bireyler modüllere internet üzerinden ulaşılabilirler.
- Basılmış modüller, eğitim kurumlarında öğrencilere ücretsiz olarak dağıtılır.
- Modüller hiçbir şekilde ticari amaçla kullanılamaz ve ücret karşılığında satılamaz.

İÇİNDEKİLER

AÇIKLAMALAR	iii
GİRİŞ	1
ÖĞRENME FAALİYETİ - 1	3
1. WEB FORMLARI	3
1.1. Sınıf (Class)	3
1.2. Nesne (Object)	4
1.3. Ad Alanı (Namespace).....	5
1.4. Kalıtım (Inheritance).....	7
1.5. ASP.NET Bildirimleri.....	7
1.6. Web Formları	7
1.7. Olaylar (Events)	11
1.8. Olay İşleme Programı (Event Handler).....	13
1.9. Web Formun Sunucuya Postalanması.....	14
1.10. IsPostBack.....	17
1.11. Görünüm Durumu (ViewState).....	19
1.12. ASP.NET Web Sitesini Yapılandırma	22
1.13. Durum Yönetimi	23
UYGULAMA FAALİYETİ	25
ÖLÇME VE DEĞERLENDİRME	26
ÖĞRENME FAALİYETİ – 2	27
2. WEB FORM ELEMANLARI.....	27
2.1. ASP.NET Sitesi Oluşturma.....	28
2.2. Web Form Oluşturma.....	34
2.3. Web Forma Kontrol Ekleme	35
2.4. Sayfayı Kaydetme	35
2.5. Web Form Elemanları (Sunucu Kontrolleri).....	36
2.6. HTML Sunucu Kontrolleri.....	36
2.7. Web Sunucu Kontrolleri	39
2.7.1. Label	39
2.7.2. TextBox	40
2.7.3. Button	41
2.7.4. LinkButton.....	42
2.7.5. CheckBox	43
2.7.6. CheckBoxList	43
2.7.7. DropDownList	45
2.7.8. ListBox	49
2.7.9. RadioButton.....	51
2.7.10. RadioButtonList.....	52
2.7.11. Table, TableRow, TableCell.....	53
2.7.12. Literal.....	55
2.7.13. Panel	56
2.7.14. AdRotator	57
2.7.15. Calendar.....	58
2.7.16. HyperLink.....	59
2.8. Geçerlilik Sunucu Kontrolleri.....	60
UYGULAMA FAALİYETİ	62

ÖLÇME VE DEĞERLENDİRME	66
ÖĞRENME FAALİYETİ - 3	67
3. ADO.NET	67
3.1. Veri (Data)	67
3.2. Veri Kaynağı (Data Source).....	67
3.3. Veritabanı Kavramları.....	69
3.4. SQL (Structured Query Language)	70
3.4.1. SELECT Deyimi.....	70
3.4.2. INSERT Deyimi	71
3.4.3. UPDATE Deyimi	71
3.4.4. DELETE Deyimi	71
3.5. ADO.NET ve Veritabanları	72
3.5.1. .NET Veri Sağlayıcısı (.NET Framework Data Provider).....	72
3.5.2. Veri Seti (DataSet).....	74
3.6. Veri Kaynağı Kontrolleri (Data Source Controls)	90
3.6.1. AccessDataSource	90
UYGULAMA FAALİYETİ	99
ÖLÇME VE DEĞERLENDİRME	101
MODÜL DEĞERLENDİRME	102
CEVAP ANAHTARLARI.....	104
KAYNAKÇA	105

AÇIKLAMALAR

KOD	481BK0092
ALAN	Bilişim Teknolojileri
DAL/MESLEK	Web Programcılığı
MODÜLÜN ADI	Etkileşimli Web Uygulamaları-3
MODÜLÜN TANIMI	Etkileşimli web uygulamaları geliştirirken kullanılan web form elemanlarının yapısının anlatıldığı öğrenim materyalidir.
SÜRE	40/32
ÖN KOŞUL	Etkileşimli Web Uygulamaları 1 modülünü bitirmiş olmak.
YETERLİK	Programlama dilinin komut yapısını tanıyarak uygulama içinde kullanmak.
MODÜLÜN AMACI	Genel Amaç Gerekli ortam sağlandığında, etkileşimli web uygulamaları geliştirirken form elemanlarını tanıyarak uygulama içinde kullanabileceksiniz. Amaçlar 1.Web form oluşturabileceksiniz. 2.Web form elemanlarını kullanabileceksiniz. 3.Veritabanı uygulamaları yapabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	Web programlama yazılımlarını çalıştırabilecek yeterlikte donanıma sahip bilgisayar.
ÖLÇME VE DEĞERLENDİRME	<ul style="list-style-type: none">➤ Her faaliyet sonrasında o faaliyetle ilgili değerlendirme soruları ile kendi kendinizi değerlendireceksiniz.➤ Modül sonunda uygulanacak ölçme araçları ile modül uygulamalarında kazandığınız bilgi ve beceriler ölçülerek değerlendirilecektir.

GİRİŞ

Sevgili Öğrenci

Güzel bir yemek gördüğümüzde beğenimizi ifade ederiz. O yemeğin masaya gelene kadar mutfakta nasıl hazırlandığını çoğunlukla düşünmeyiz. Yemeğin malzemelerinin yıkanması, soyulması, doğranması, uygun ölçülerde tencereye konulması, yemeğin kıvama kadar pişirilmesi, sosunun hazırlanması, masanın hazırlanması vs. gibi zahmet veren işleri yapan birileri vardır mutfakta. İşte sizler de kullanışlı, beğenilen bir web sitesinin mutfağında çalışanlarıdır. Web sitesinizi hem sizin beğenmeniz hem de kullananların beğenmesi için her türlü zahmete girmeye hazır mısınız?

Daha önceki modüllerde etkileşimli web uygulamaları geliştirmek için gerekli temel kavramları ve program kodu yazmak için kullanılacak dilin yapısını öğrendiniz. Bu modülde ise sınıflar, nesnelere, web formları, veritabanı erişimi kavramlarını öğreneceksiniz.

Yeni karşılaştığımız konuların mantığını kavramaya çalışmanız, gerektiğinde araştırmalar yapmanız konuyu sizin açınızdan daha anlaşılır hâle getirecektir. Verilen örneklere ek olarak sizler de örnekler geliştirmeye çalışınız.

ÖĞRENME FAALİYETİ-1

AMAÇ

Web formları oluşturabileceksiniz.

ARAŞTIRMA

Günlük yaşamımızda, çevremizdeki varlıkları ortak özelliklerine göre çeşitli sınıflara ayırırız. Örneğin, serçe, güvercin, kartal, papağan, kanarya gibi canlıları kuşlar olarak sınıflandırırız. Bu tip sınıflandırmaları bilgisayar dünyası da dâhil olmak üzere her alanda gerçekleştiririz. Sizler de yukarıdaki örnekte verildiği gibi yaşantımızda varolan sınıflandırmalara 5 örnek veriniz.

1. WEB FORMLARI

Web formları web uygulamalarının vazgeçilmezidir. Web formları, web uygulamaları için bir arayüzdür. Web uygulamasının etkileşimli olması için, yani uygulamanın kullanıcının isteğine cevap verebilmesi için gerekli olan kullanıcı bilgileri web formlar vasıtasıyla alınır. Web formlarını form verisinin geçerliliğini denetlemek (onaylamak) ve veritabanında tutulan veriyi düzenlemek ve göstermek gibi eylemleri gerçekleştirmek için kullanabiliriz.

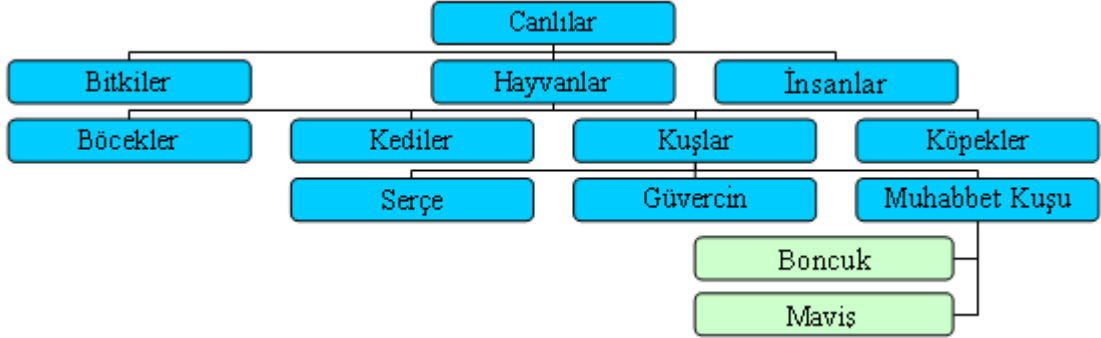
Bir ASP.NET sayfası, kullanıcının gördüğü form ve formun çalışma mantığını oluşturan program kodlarından oluşur. Bu yönüyle ASP.NET sayfasına genel olarak web form denilebilir. Bu öğrenme faaliyetinde web formlarıyla ilişkili kavramlar açıklanacaktır. Öğrenme Faaliyeti-2’de Dreamweaver MX2004 kullanılarak web kontrolleri tanıtılacaktır, Öğrenme Faaliyeti-3’te ise veritabanı uygulamalarına yer verilecektir.

1.1. Sınıf (Class)

Programlama dünyasında sınıf kavramına geçmeden önce konunun daha anlaşılır olması için doğal hayattan bir örnek verelim. Dünya ağaç, bina, yol, araba, kuş, dolap gibi canlı ve cansız varlıklardan oluşmuştur. Bizler bu varlıkları ortak özelliklerine göre sınıflara ayırırız.

Canlı varlıkları ele alalım. Biyolojide daha farklı ve ayrıntılı olmasına karşın biz bahsettiğimiz sınıflandırmayı genel olarak Şekil 1.1’deki gibi yapabiliriz. Canlılar sınıfı bitkiler, hayvanlar, insanlar olarak üç alt sınıfa ayrılmıştır. Canlıları oluşturan bitkiler, hayvanlar ve insanlar doğar, yaşar ve ölürlür. Benzer özelliklere sahip olan bitkiler, hayvanlar ve insanlar sınıfları, kimi farklılıklarına göre farklı alt sınıflara sahiptirler.

Hayvanlar alt sınıfı kendi arasında böcekler, kediler, kuşlar, köpekler şeklinde alt sınıflara ayrılmıştır. Kuşlar alt sınıfı da kendi içinde serçe, güvercin, muhabbet kuşu olarak alt sınıflara ayrılmıştır. Tüm kuşlar diğer hayvanlar gibi hareket eder, beslenir. Bu kuş türlerinin hepsi kuşlar sınıfının ortak özelliklerine sahiptir. Hepsinin gagası, kanatları, sesleri vardır. Ayrıca istisnalar dışında hepsi uçar, öter, ürer. Kuşlar, karşılaştıkları çeşitli olaylara tepki verir. Örneğin, kendisini avlamaya çalışan birisini gördüğünde tepki olarak kaçmaya başlar.



Şekil 1.1: Genel olarak canlıların sınıflandırılması

Programcılıktaki sınıflandırma mantığı da doğal yaşamımızdaki sınıflandırma mantığıyla benzerlik göstermektedir. Canlıların yaptığı hareket etmek, beslenmek, üremek, ötmek eylemleri programlamada metod (Method) kavramına karşılık gelmektedir. Kuşlara ait olan gaga, kanat, ses vs. gibi özellikler de programlamada özellik (property) kavramına eşittir. Kuşun avlanmaya çalışılması olay (event), kuşun avlanmaya karşı yaptıklarının tümü olay işleme programı (event handler) kavramına denk düşmektedir.

1.2. Nesne (Object)

Eskiden programcılar geliştirdikleri programın her parçasını kendileri yazarlardı. Sıfırdan pencereler, butonlar (düğme), menüler oluştururlardı. Sonradan her programda ortak olarak kullanılacak program parçaları için farklı bir yöntem izlenmeye başlandı. Programcılar sınıf (class) yazarak, bu sınıftan da nesnelere türeterek aynı işlem için tekrar tekrar kod yazmaktan kurtulmuşlardır. Artık programına buton nesnesi eklemek isteyen programcı, ilgili sınıfla programatik olarak temasa geçip bir buton nesnesini programına dâhil etmektedir (**nesneye yönelik programlama**). Programcı, bu butonun üzerindeki yazıyı, yazının tipini, rengi, butonun büyüklüğünü istediği gibi ayarlayabilmektedir.

Bu örneği genişletecek olursak programcı kullanıcının kullanıcı adını ve şifresini isteyen bir form oluşturacak diyelim. Programcı, label sınıfından iki label nesnesini, textbox sınıfından iki textbox nesnesi, button sınıfından bir button nesnesini programına dâhil edecektir (Resim 1.1).



Resim 1.1: Kullanıcı adı ve şifre penceresi için gerekli nesnelere eklenmesi

Ardından nesnelerin text özelliklerinde değişiklik yapılarak pencerenin son şekli verilecektir.



The image shows a web form with two text input fields. The first field is labeled 'Ad' and the second is labeled 'Soyad'. To the right of the 'Soyad' field is a button labeled 'Gönder'.

Resim 1.2: Nesnelerin özelliklerinde gerekli değişikliklerin yapılması

Kısacası sınıflar nesnelerin kullanılması için bir altyapı oluştururlar. Nesneler, sınıflara dayandırılarak türetilir. Uygulamamızda bir nesne oluşturup kullanmak istiyorsak buna ait sınıfla temasa geçip nesneyi oluşturabiliriz. Nesneler, özelliklerini ait oldukları sınıftan alır.

Canlılar ile ilgili örnekte Boncuk ve Maviş isimli muhabbet kuşlarını nesne olarak düşünebiliriz. Bu iki kuş muhabbet kuşları sınıfının tüm özelliklerini miras almıştır (**inheritance**). Her iki kuş da bağlı buldukları muhabbet kuşu sınıfa ait gaga, kanat ve ses vs. özelliklere sahiptir ve o sınıfa ait hareket etmek, beslenmek, üremek, ötmek eylemlerini gerçekleştirmektedir. Programlama dünyasında da nesneler türetildikleri sınıfın özellik ve metodlarını alır.

1.3. Ad Alanı (Namespace)

Canlıların sınıflandırılması örneğine dönelim. Canlıları çeşitli özelliklerine göre gruplandırabiliriz. Örneğin, hayvanlar ve insanlar sınıf ve alt sınıflarının tümünü hareket edebilenler, bitkiler sınıf ve alt sınıflarını hareket edemeyenler olarak gruplandırabiliriz. Belirli ortak özelliklerine göre sınıfların gruplandırılması .NET dünyasında ad alanları (**namespace**) kavramıyla eşleştirilebilir. Ad alanları, benzer görevleri yerine getiren sınıfların gruplandırılmasından oluşmaktadır. Kimi kaynaklarda namespace kavramı için **ad uzayı, isim uzayı** ifadeleri kullanılmaktadır.

Sınıfları dolayısıyla nesnelere içinde barındıran ad alanlarının programa dâhil edilmesi gerekmektedir. ASP.NET sayfalarına otomatik olarak eklenen bazı ad alanları aşağıda listelenmiştir:

```
System
System.IO
System.Collection
System.Web
System.Web.UI
System.Web.UI.HtmlControls
System.Web.UI.WebControls
```

Kimi ad alanlarını ise programa bir bildirim ile dâhil etmeniz gerekmektedir. Bu işleme import adı verilir. Bu amaçla **@import** bildirimini kullanılır. Ad alanlarının import edilmesi her seferinde nesnelere tam ismini yazmak zorunluluğunu ortadan kaldırır.

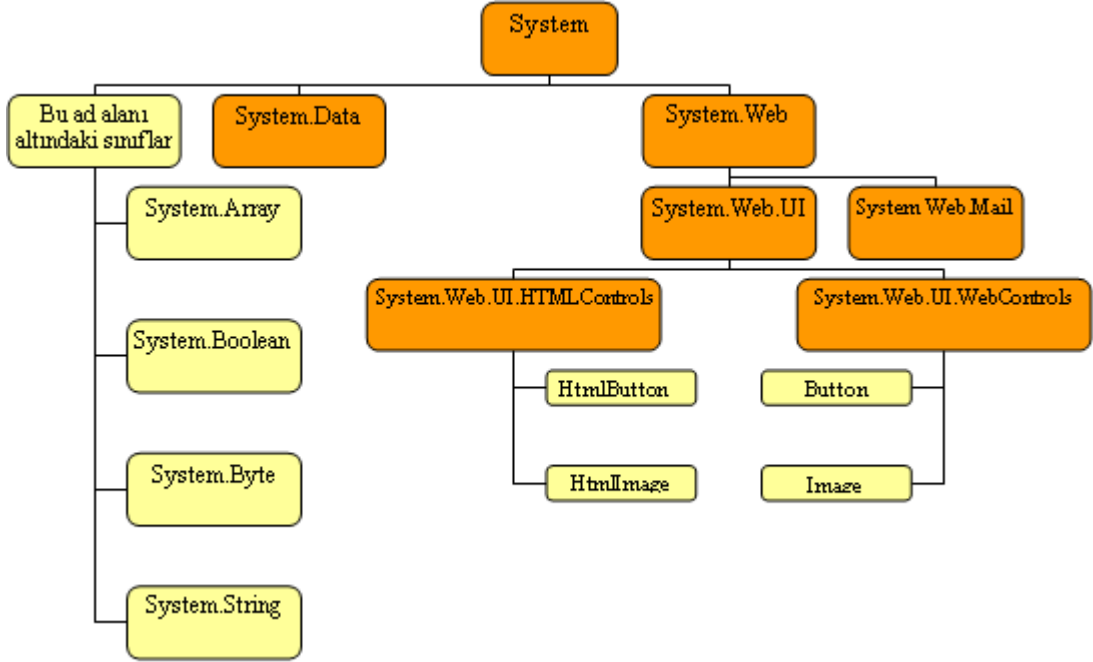
import.aspx (Bir ad alanının import edilmesi)

```
<%@ Import Namespace="System.Drawing" %>
```

```

<script runat="server">
Sub Page_Load()
    Label1.ForeColor = Color.Blue
End Sub
</script>

```



Şekil 1.2: .NET Framework'te ad alanları (turuncu renkli) ve sınıflar (sarı renkli) arası ilişki

```

<body>
<form runat="server">
    <asp:Label ID="Label1" Text="Renk" runat="server"></asp:Label>
</form>
</body></html>

```

UYGULAMA

1. Sayfayı import satırını silerek import2.aspx adıyla kaydediniz. Sayfayı çağırıp sonucu yorumlayınız.
2. Import2.aspx sayfasında Page_Load içindeki satırı aşağıdaki şekilde değiştirip sayfayı çağırınız. Sonucu yorumlayınız.

```
Label1.forecolor = System.Drawing.Color.Blue
```

.Net Framework içerisinde yer alan tüm ad alanları ve sınıflarını quickstarts.asp.net/QuickStartv20/util/classbrowser.aspx adresinden veya **Lutz Roeder's .NET Reflector** programından inceleyebilirsiniz.

1.4. Kalıtım (Inheritance)

Programlamada hazır kodların tekrar kullanımı kalıtım (**miras alma = inheritance**) yoluyla gerçekleştirilir. Kalıtımla türetilen sınıf için temel teşkil edecek sınıf tanımlanır. Türetilmiş sınıf (**derived class**), temel sınıfın (**base class**) özelliklerini (properties), metotlarını (methods) kalıtımla alıp genişletebilir. Temel sınıftan kalıtımla bir sınıf türetilmek istendiğinde **inherits** ifadesi kullanılır.

1.5. ASP.NET Bildirimleri

@Page bildiri (yönerge, directive) .aspx sayfalarına eklenir ve ASP.NET ayrıştırıcı (parser) ve derleyicisi (compiler) tarafından kullanılan sayfa özelliklerini belirler. Daha önceki uygulamalarda bu bildirim **language** ve **codepage** özelliklerini kullanırdınız.

Bildirim **CodeBehind** özelliği ASP.NET sayfalarının görsel yüzü ile arka planda çalışan kodların ayrı sayfalarda tutulmasını sağlar. Örneğin, hazırlanan sayfanın istemcinin karşısında göreceği arayüzü **sayfa1.aspx** dosyasında, sayfanın programlama mantığı **sayfa1.aspx.vb** (Programlama dili olarak C Sharp kullanılıyorsa **sayfa1.aspx.cs**) dosyasında tutulur.

sayfa1.aspx	sayfa1.aspx.vb
<pre><%@page language="VB" runat="server" %> <html><body> <form runat="server"> <asp:label id="label1" runat="server"/> </form> </body></html></pre>	<pre><script runat="server"> sub page_load(sender as object, e as eventargs) label1.text = "Merhaba" end sub </script></pre>

Şekil 1.3: Code-Behind

1.6. Web Formları

Web formlarının oluşturulmasındaki amaç, web uygulamasında kullanıcıya gözüken elemanları sunucunun kontrolüne almak, böylelikle web uygulama geliştiricisinin uygulama üzerindeki kontrolünü artırmaktır.

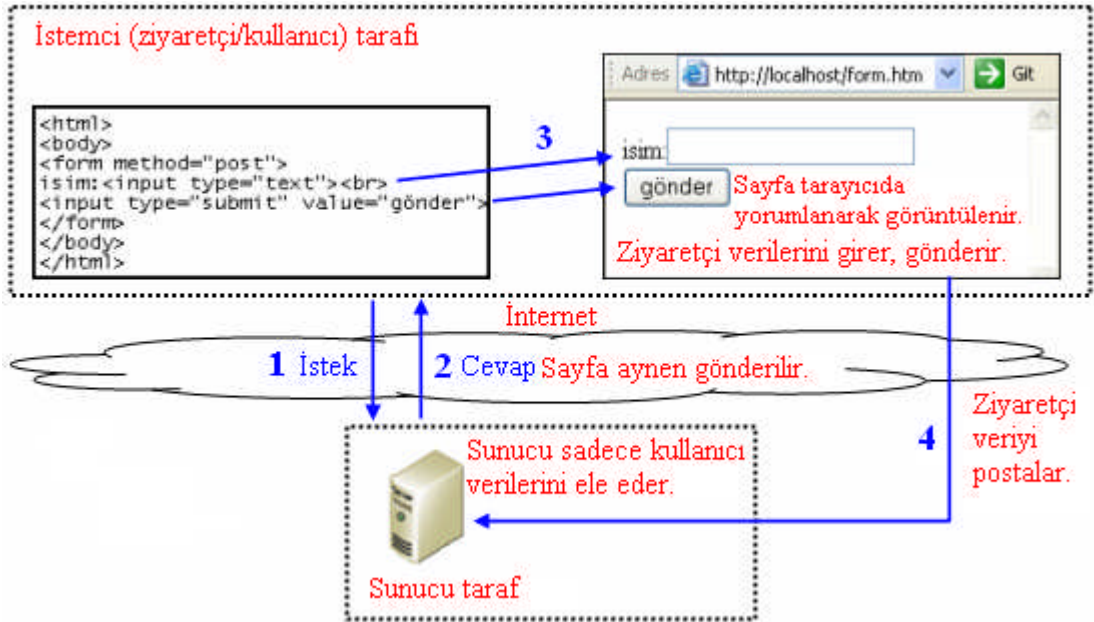
Bildiğiniz gibi formlar ziyaretçinin çeşitli şekillerde veri girmesini sağlar. Örneğin, ziyaretçi ad soyad bilgilerini metin kutusuna girer, tercihlerini onay kutusu (checkbox) kullanarak işaretler.

Web formlarını anlatmadan önce HTML formlarının nasıl çalıştığını hatırlayalım. HTML sayfaları sunucu tarafından istemciye gönderilir. Eğer sayfada form elemanlarını içeren etiketler varsa bu etiketler tarayıcı tarafından yorumlanır. Böylelikle form, sayfada görüntülenir. Görüldüğü üzere tüm işlemler istemci tarafında gerçekleşir, sunucu sadece istemcinin istediği sayfayı gönderir. Formların oluşturulması tamamen tarayıcı tarafından gerçekleştirilir. Ziyaretçinin forma girdiği veriler sunucuda işlendikten sonra sonuçlar istemciye gönderilir. Sunucunun buradaki rolü, kullanıcıdan gelen bilgileri işlemektir.

Basit bir HTML formun oluşturulması aşağıdaki gibidir.

```
<html><body>
<form method="post">
  isim: <input type="text"><br>
  <input type="submit" value="gönder">
</form>
</body></html>
```

Yukarıdaki örnekte **post** metodu ile forma girilen veriler sunucuya gönderilir. Sunucu, sunucu taraflı kodlar vasıtasıyla verileri işler. Sunucu, kodlarla kısıtlı olarak form elemanlarının özelliklerini (label elemanının eni, yüksekliği gibi) değiştirebilir.



Resim 1.3: HTML formlarının istemci-sunucu modeline göre işlenmesi

Web formlarında durum farklıdır. Web formları tamamıyla sunucuda oluşturulduktan sonra HTML etiketlerine dönüştürülerek istemciye gönderilir. İstemciye kullanıcı tarafındaki değişimleri takip eden kodlar da gönderilir. Bu kodlar genellikle form elemanlarının durumunu takip edip sunucuya bildiren istemci taraflı kodlardır. Web form elemanları tamamen sunucuya ait elemanlar olduğundan kontrolü, sunucuya dolayısıyla web programcısına aittir. Sunucu da, web uygulamasının çalışması sırasında her türlü işlemi web formları üzerinde gerçekleştirebilir ve işlem sonuçları HTML çıktısı olarak istemciye gönderilir. Basit bir web formun oluşturulması aşağıdaki gibidir.

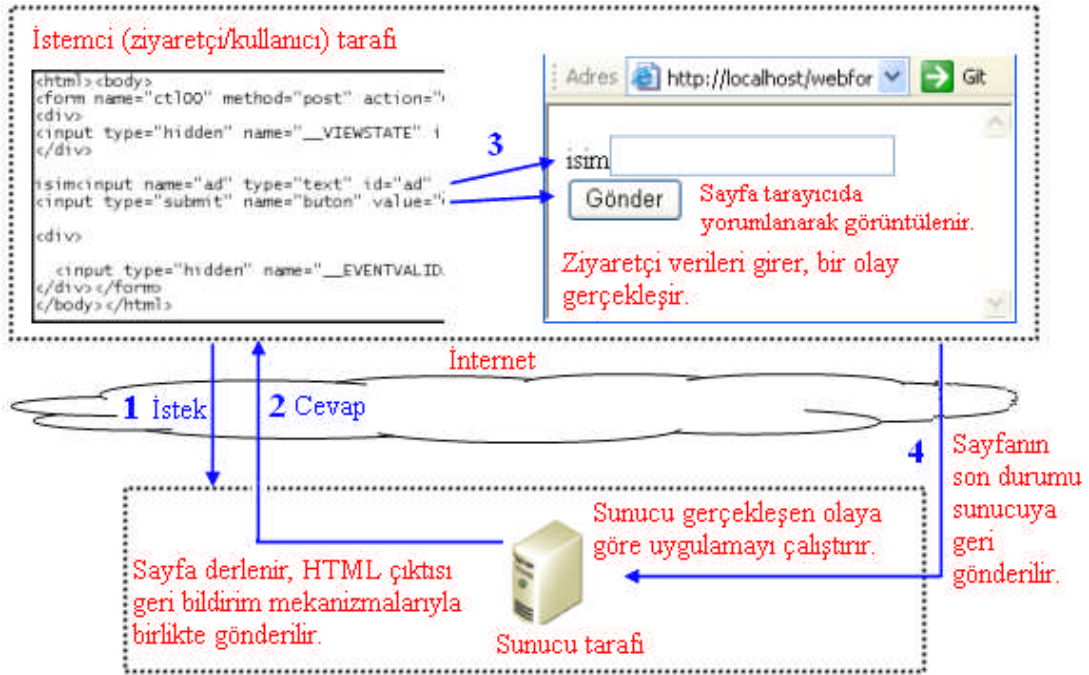
Webform.aspx

```
<script language="vb" runat="server">
sub gonder(sender as object, e as eventargs)
```

```

    response.write("Merhaba")
end sub
</script>
<html><body>
<form runat="server">
    isim<asp:textbox id="ad" runat="server"/>
    <asp:button id="buton" text="Gönder" onclick="gonder"
        runat="server"/>
</form></body></html>

```



Resim 1.4: Web formlarının istemci-sunucu modeline göre işlenmesi

Peki sunucu, istemci bilgisayardaki formları nasıl kontrol eder? Örneğin, formda bir seçim yapıldığında sunucunun bu seçimden nasıl haberi oluyor? Form elemanlarındaki her değişime **olay** (event) adı verilir. Sunucu formdaki değişimlerde sayfanın tümünü değil, olaydan etkilenen kısımları işler. Sunucu, istemci bilgisayara gönderdiği gizli form elemanlarıyla ve istemci tarafı kodlarla istemci tarafındaki değişimlerden haberdar olur. Böylelikle sunucu, web formları vasıtasıyla uygulamanın her aşamasını kontrol altında tutmaktadır.

Web uygulamalarında web formlarının iki kısımdan oluştuğunu görürüz: Birincisi, sayfada görülecek elemanlar ve bu elemanlarda oluşan olay sonucunda çalıştırılacak alt programlar.

Daha önce de gördüğümüz gibi web formları aşağıdaki şekilde oluşturulur:

<asp:kontrol ismi id="kontrol kimliği" runat="server">

asp:kontrol ismi = Oluşturulacak sunucu kontrolünün (form veya form elemanı) isminin yazıldığı kısımdır.

id="kontrol kimliği" = Kontrolü diğer kontrollerden ayıran kimlik bilgisi burada verilir.

runat="server" = Kontrolün sunucuda çalışan bir kontrol olduğunu gösterir.

Yukarıdaki satıra kontrole özgü başka bölümler de eklenebilir. Ayrıca kontrol bir olayla ilişkilendirilmişse o olayın ismi ve olayın oluşması durumunda işlenecek alt programın ismi de bu satıra eklenir. Aşağıda buna uygun örnek verilmiştir:

<asp:button id="btn1" text="Onaylıyorum" onclick="onay" runat="server"/>

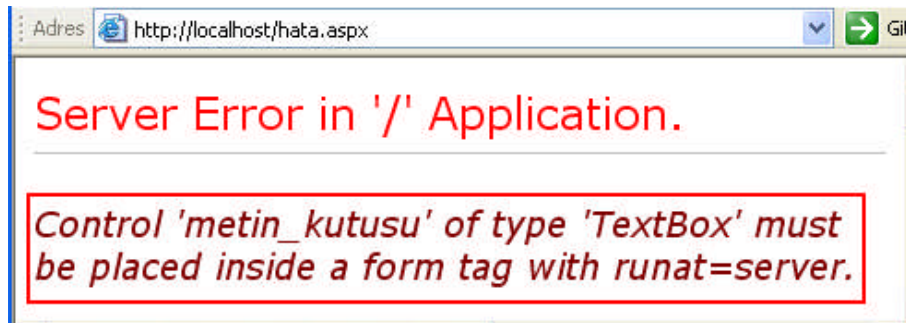
onclick="onay" = Buton tıklandığında (yani click olayı oluştuğunda) onay isimli alt program (olay işleme programı = event handler) çalışır.

Her formun bir ismi olmalıdır. Siz bir isim belirtmezseniz ASP.NET forma bir isim otomatik olarak atar. Siz **id** özelliğini kullanarak formun ismini belirleyebilirsiniz.

Eğer metin kutusu gibi barındırcı olmayan bir kontrol, form etiketi dışına yerleştirilirse **HttpException** istisnası oluşur. Derleme zamanında (**compile time**) kontrol yapılmadığından çalışma anında (**run time**) böyle bir istisna (**exception**) oluşur. Aşağıdaki hata.aspx sayfasında metin kutusu form etiketi içerisine yerleştirilmediğinden hata oluşmuştur.

Hata.aspx

```
<html><body>
<asp:textbox id="metin_kutusu" runat="server"/>
</body></html>
```



Resim 1.5: Metin kutusunun form etiketi içerisine yerleştirilmemesinden kaynaklanan hata:Textbox tipindeki metin kutusu kontrolü runat=server ifadesini içeren form içine yerleştirilmelidir.

1.7. Olaylar (Events)

Web sitesinin kullanıcıları sunucu kontrolleri üzerinde işlem yaptığında olay meydana gelir. Örneğin, kullanıcı, düğmeye tıkladığında **click** olayı gerçekleşir. Olay gerçekleştiğinde hangi işlemlerin yapılacağı ise olay işleme programında belirtilir. Olay ifadeleri “on” ifadesi ile başlar ve olayı ifade eden bir isimle devam eder. Görsel bir editörde kod yazımı sırasında bu olaylar listelenir.

Aşağıda **button** ve **textbox** kontrollerine ait olaylar listelenmiştir. Kimi olayların her iki kontrol için ortak, kimi olayların da sadece o kontrole özgü olduğuna dikkat ediniz. İstemci bilgisayarda yapılan bir işlem sonucu oluşan olaya bağlı alt program sunucuda çalıştırılır. Çalışma sonucu istemciye aktarılır.

Button (Buton) kontrolü olayları	Textbox (Metin kutusu) kontrolü olayları
OnDataBinding	OnDataBinding
OnDisposed	OnDisposed
OnInit	OnInit
OnLoad	OnLoad
OnPreRender	OnPreRender
OnUnLoad	OnUnLoad
OnClick	OnTextChanged
OnCommand	

Olaya bağlı alt programların çalışmasını daha önceki örneklerde gördünüz. Düğmeye her tıkladığında sayıyı bir artıran sayfa örneğiyle olaya bağlı alt programların çalışmasını tekrar gözden geçirelim:

sayiArtir.aspx

```
<%@ page language="VB" debug="true" %>

<script runat="server">
sub button1_click(sender as object, e as eventargs)
    labell1.text=convert.toString(convert.toInt32(labell1.text) + 1)
end sub
</script>
<html><body>
<form runat="server">
    <asp:button id="button1" text="Artır" onClick="button1_click"
        runat="server" /><br>
    <asp:label id="labell1" text="0" runat="server" />
</form></body></html>
```

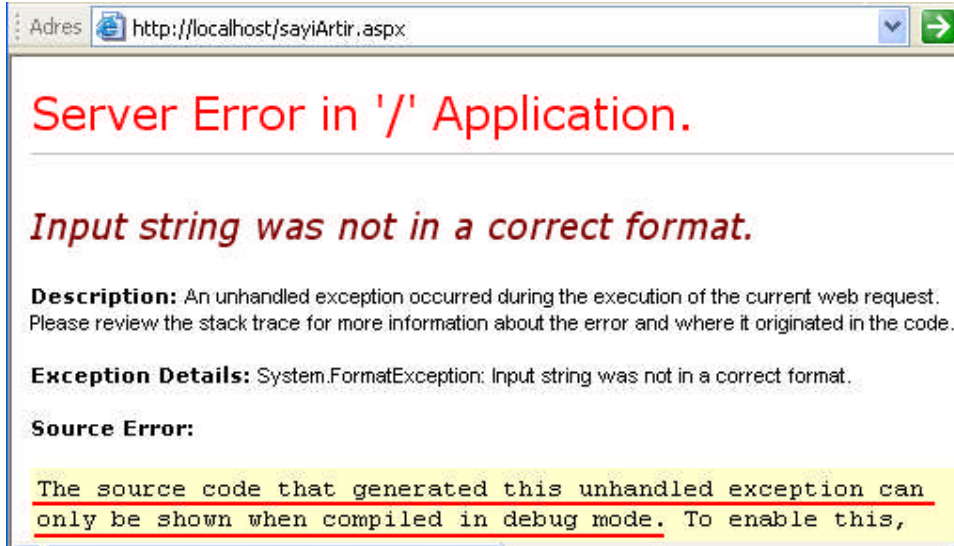
Sayfada düğmeye tıklandığında olay gerçekleşmiş, sayfa sunucuya geri postalanmıştır (**Postback**). Sunucunun yaptığı işlemlerin sonucu yine aynı sayfada görüntülenmiştir. Düğmeye tıklandığında yapılacak işlemler **button1_click** alt programında belirtilmiştir. Olay gerçekleştiğinde işlenecek kodları içeren alt programa **event handler (olay işleme programı, olay işleyici)** adı verilir.



Resim 1.6: SayiArtir.aspx sayfasının çalışması

Page bildiriminde **debug="true"** ile kodun **debug (Hata ayıklama)** modunda derlenmesi sağlanmıştır. Bu mod, sayfa geliştirilirken sayfadaki hatanın hangi satırdan kaynaklandığının görüntülenmesini sağlar. Sayfaya son şekli verilip İnternette yayınlanmadan önce bu mod pasif hâle getirilmelidir. Bu özelliğin ne işe yaradığını şu şekilde görelim: **Page** bildiriminden **debug="true"** ifadesini kaldırarak (veya **debug="false"** şeklinde değiştirerek) **label** satırındaki **text="0"** ifadesini silip sayfayı hata üretecek hâle getirelim ve sayfayı tekrar çalıştıralım.

Resim 1.7’de görüldüğü gibi hatalı olan veya hatayla ilişkili satır görüntülenmemiştir. **Debug="True"** ifadesini ekleyerek sayfa tekrar çağrıldığında ise sayfadaki hatalı veya hatayla ilişkili satır belirtilmiştir (Resim 1.8).



Resim 1.7: Debug modu aktif değilken hatalı satırın görüntülenmemesi

Source Error:

```
Line 3: <script runat="server">
Line 4: sub button1_click(sender as object, e as EventArgs)
Line 5: Label1.Text=Convert.ToString(Convert.ToInt32(Label1.Text) + 1)
Line 6: end sub
```

Resim 1.8: Debug modu aktif iken hatalı satırın görüntülenmesi

5.satırda **label**'in değerini 1(bir) artıran kod bulunmaktadır. Kod satırında iki defa tip dönüşümü gerçekleştirilmiştir. **Convert.ToInt32(Label1.Text)** koduyla **label**'in **text** değeri **integer** tipine dönüştürülmüştür. Bu sayısal değere bir eklendikten sonra ortaya çıkan yeni sayısal değer **Convert.ToString** koduyla **string** tipine dönüştürülmüştür.

1.8. Olay İşleme Programı (Event Handler)

ASP.NET'te olayları işlemek için olay işleme programları kullanılır. Her ASP.NET nesnesi özel olay işleyici olarak atanmış alt programlara sahiptir. Örneğin, Page nesnesi olayları için genel olarak Page_Init, Page_Load, Page_DataBind, Page_PreRender, and Page_Unload alt programları vardır.

Page nesnesi dışındaki nesnelere oluşan olayları işleyecek alt program 2 yöntem kullanılarak belirtilir. Birinci yöntem, kontrolün sözdiziminde (sentaks, syntax) olay için alt programı atamaktır. sayiArtir.aspx sayfasında button nesnesinde bu yöntem kullanılmıştır.

```
<asp:button id="button1" text="Artır" onClick="button1_click"
runat="server" />
```

İkinci yöntem, komut kullanarak olaya alt program atamaktır. VB'de bu işlem için kullanılan komut **AddHandler** komutudur. Bu komut, tasarımcının tamamen sayfa tasarımına web programcısının ise kodlamaya odaklanmasını mümkün kılar. SayiArtir.aspx sayfasında button1_click alt programından önce aşağıdaki satırları ekleyerek ve

<asp:button... satırından onClick="button1_click" ifadesini çıkararak sayfayı **sayiArtir2.aspx** sayfası olarak kaydedelim. İkinci yöntemi bu sayfa üzerinden inceleyelim.

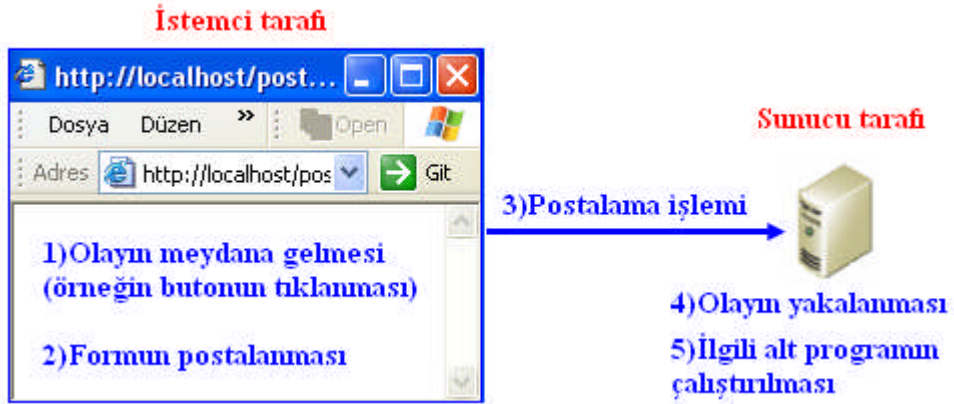
```
sub Page_Load()  
AddHandler button1.click, AddressOf Me.button1_click  
end sub
```

AddHandler komutu ile **button1** kontrolünün **click** olayı **button1_click** alt programına atanmıştır.

1.9. Web Formun Sunucuya Postalanması

İstemci bilgisayarda web formlarıyla çalışma esnasında olayın oluşmasıyla postalama işlemi (postback) başlar. Postalanan veriler sunucuda işlenerek tekrar istemciye gönderilir. ASP.NET'te web formları farklı bir sayfaya değil kendine geri postalanır.

İstemci bilgisayarda görüntülenen bir web form sunucuya postalandığında web form postback modundadır. Dolayısıyla kullanıcı web formu ilk kez açıyor web form postback modunda değildir.



Resim 1.9: Web formların postalanması

İstemci bilgisayardan sunucuya veri postalama iki yolla gerçekleşir:

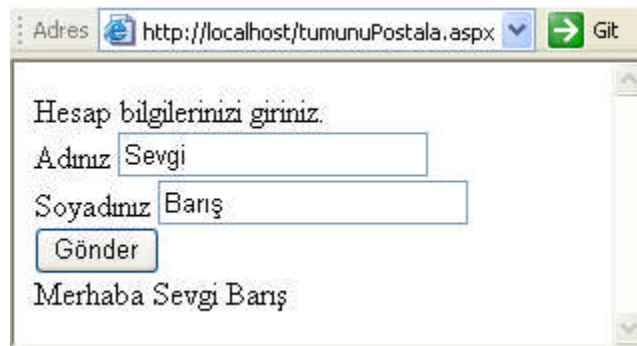
- Formdaki tüm kontrollere değer girildikten sonra form postalanır. Örneğin, bir e-posta hesabı alırken tüm bilgiler girildikten sonra formun postalanması gibi.
- Form içinde bir elemandaki değişimde verilerin hemen sunucuya gönderilmesidir. Örneğin, e-posta hesabı alırken ülke kısmında tüm ülkeler listelenmiştir. Buradan Türkiye seçildiğinde form sunucuya postalanır, sunucu Türkiye illerini gösteren kısmı istemciye gönderir ve site ziyaretçisi istediği ili seçebilir.

Web formlarının postalanma şekillerinin daha iyi anlaşılması için aşağıdaki **tumunuPostala.aspx** ve **hemenPostala.aspx** sayfalarını inceleyelim:

tumunuPostala.aspx

```
<script runat="server">
sub goruntule(sender as object, e as eventargs)
    label4.text = "Merhaba " & ad.text & " " & soyad.text
end sub
</script>
<html><body>
<form runat="server">
    <asp:label id="label1" text="Hesap bilgilerinizi giriniz."
        runat="server" /><br>
    <asp:label id="label2" text="Adınız" runat="server" />
    <asp:textbox id="ad" onTextChanged="goruntule"
        runat="server" /><br>
    <asp:label id="label3" text="Soyadınız" runat="server" />
    <asp:textbox id="soyad" onTextChanged="goruntule"
        runat="server" /><br>
    <asp:button id="buton1" text="Gönder" onClick="goruntule"
        runat="server" /><br>
    <asp:label id="label4" runat="server" />
</form></body></html>
```

Sayfada **Gönder** düğmesine tıklandığında, tüm bilgiler sunucuya gönderilmiş, bilgiler sunucuda işlendikten sonra sonuç tekrar istemciye gönderilmiştir. Burada metin kutusundaki değer değiştiği hâlde (**onTextChanged**) form sunucuya geri postalanmamıştır.

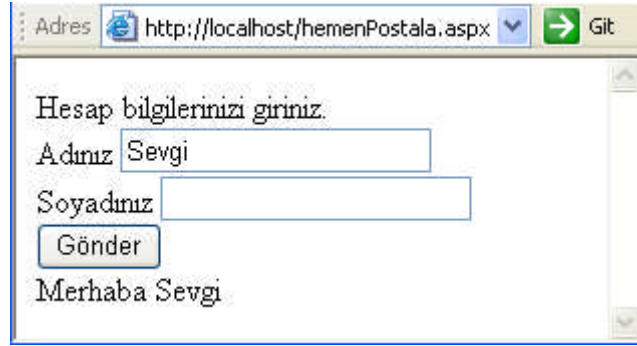


Resim 1.10: Gönder düğmesiyle formun postalanması

Form elemanlarından birindeki değişiklik üzerine verilerin sunucuya gönderilmesi istendiğinde o form elemanının bildirimine **AutoPostBack = true** ifadesi eklenmelidir. **AutoPostBack** özelliği formun **otomatik geri postalanması** anlamına gelir. Eğer değeri **true** olursa form elemanındaki değişiklikte veriler sunucuya postalanır. Bu işlemle ilgili olarak da **hemenPostala.aspx** sayfasını inceleyelim.

hemenPostala.aspx

```
<script runat="server">
sub goruntule(sender as object, e as eventargs)
    label4.text = "Merhaba " & ad.text & " " & soyad.text
end sub
</script>
<html><body>
<form runat="server">
    <asp:label id="label1" text="Hesap bilgilerinizi giriniz."
        runat="server" /><br>
    <asp:label id="label2" text="Adınız" runat="server" />
    <asp:textbox id="ad" onTextChanged="goruntule"
        autoPostBack="True" runat="server" /><br>
    <asp:label id="label3" text="Soyadınız" runat="server" />
    <asp:textbox id="soyad" onTextChanged="goruntule"
        autoPostBack="True" runat="server" /><br>
    <asp:button id="buton1" text="Gönder" onClick="goruntule"
        runat="server" /><br>
    <asp:label id="label4" runat="server" />
</form></body></html>
```



Resim 1.11: Metin kutusundaki değişiklik üzerine formun sunucuya gönderilmesi

Sayfada metin kutusuna değer girilirken herhangi bir yolla o kontrolden ayrıldığımızda (diğer bir kontrole geçtiğimizde, enter tuşuna bastığımızda veya fare ile herhangi bir yere tıkladığımızda) kontrolde değişiklik gerçekleştirmiş oluruz. Bu değişiklik olduğu anda form sunucuya geri postalanır.

Kimi durumlarda tüm form elemanlarında işlem yaptıktan sonra formu göndermek kimi durumlarda da bir form elemanındaki değişimde verileri göndermek tercih edilir.

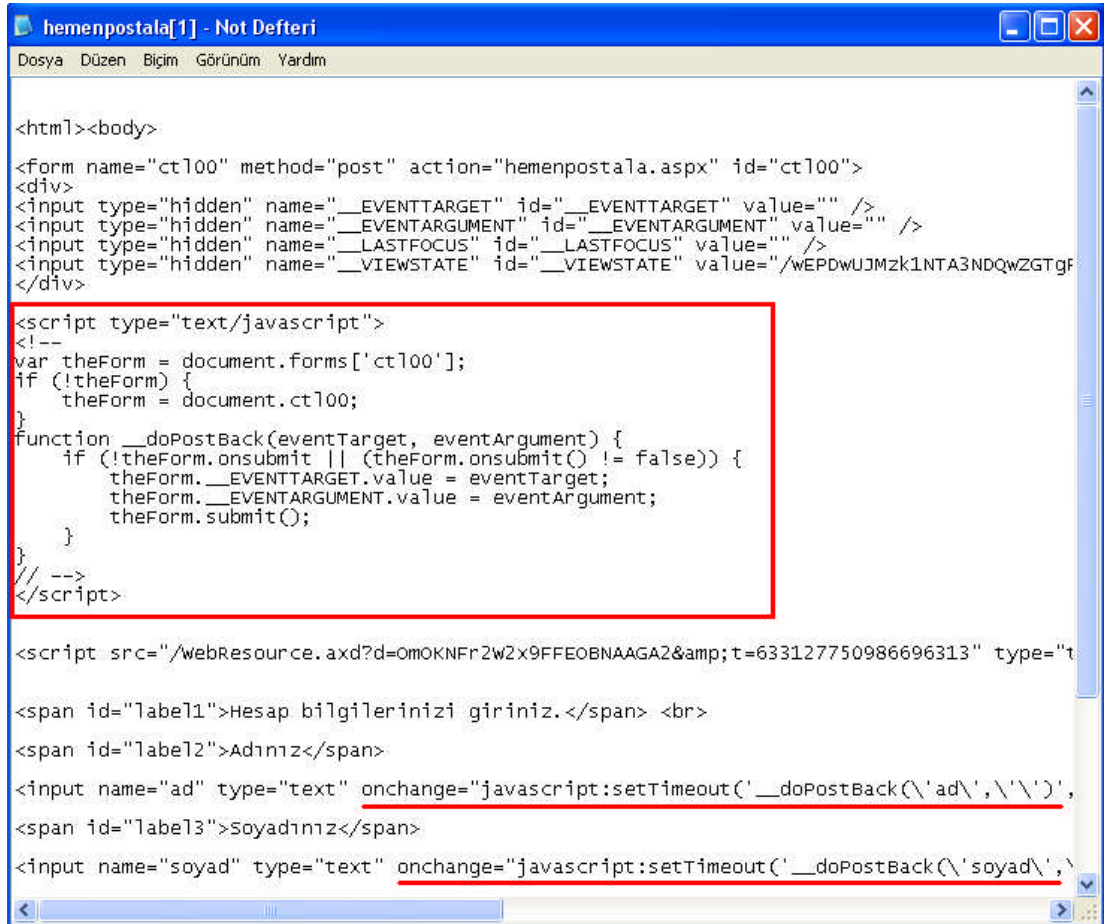
Bir form elemanındaki değişime göre diğer form elemanlarını oluşturduğumuz uygulamalarda; tek bir kontroldeki değişikliğe göre formu postalamak uygundur.

Bir kontroldeki deęişiklik üzerine yapılan postalama işleminde sunucu bilgisayar kontroldeki deęişiklikten nasıl haberdar olmaktadır? Bir başka deyişle, istemci bilgisayarda bulunan kontroldeki deęişiklikten sunucu nasıl haberdar oluyor? Cevap istemci bilgisayarda çalışan kodlardır (script). Kullanılan kodlar Javascript kodlarıdır (Resim 1.12).

Formdaki kontrolden ayrıldığımız, javascript kodları tarafından algılanır, bu durum sunucuya bildirilir. Bu kodların istemci bilgisayara gönderilmesi işlemini otomatik olarak ASP.NET yapmaktadır.

1.10. IsPostBack

ASP.NET’te sayfanın ilk defa mı yüklendiğini yoksa geri mi postalandığını (Postback) tespit etmek amacıyla **IsPostBack** özelliği kullanılır. IsPostBack özelliği, **Page** nesnesinin bir özelliğidir. Bazı işlemlerin sadece sayfa ilk yüklendiğinde çalışmasını sağlamak amacıyla kullanılır.



```
hemenpostala[1] - Not Defteri
Dosya Düzen Biçim Görünüm Yardım

<html><body>
<form name="ctl00" method="post" action="hemenpostala.aspx" id="ctl00">
<div>
<input type="hidden" name="__EVENTTARGET" id="__EVENTTARGET" value="" />
<input type="hidden" name="__EVENTARGUMENT" id="__EVENTARGUMENT" value="" />
<input type="hidden" name="__LASTFOCUS" id="__LASTFOCUS" value="" />
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE" value="/wEPDwUJMzk1NTA3NDQwZGtGf
</div>

<script type="text/javascript">
<!--
var theForm = document.forms['ctl00'];
if (!theForm) {
    theForm = document.ctl00;
}
function __doPostBack(eventTarget, eventArgument) {
    if (!theForm.onsubmit || (theForm.onsubmit() != false)) {
        theForm.__EVENTTARGET.value = eventTarget;
        theForm.__EVENTARGUMENT.value = eventArgument;
        theForm.submit();
    }
}
// -->
</script>

<script src="/WebResource.axd?d=0mOKNFr2w2x9FFEOBNAAGA2&amp;t=633127750986696313" type="t

<span id="label1">Hesap bilgilerinizi giriniz.</span> <br>
<span id="label2">Adınız</span>
<input name="ad" type="text" onchange="javascript:setTimeout('__doPostBack(\'ad\',\'\')',
<span id="label3">Soyadınız</span>
<input name="soyad" type="text" onchange="javascript:setTimeout('__doPostBack(\'soyad\',\'


```

Resim 1.12: Sayfanın kaynak kodundaki Javascript ifadeleri

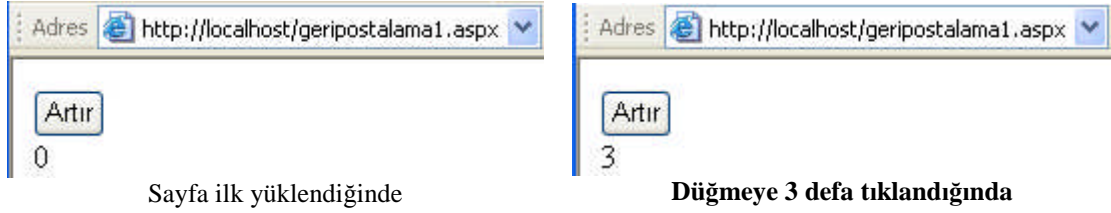
geriPostalama1.aspx

```
<script runat="server">
```

```

sub page_load(source as object, e as eventargs)
    if not page.ispostback then
        labell1.text="0"
    end if
end sub
sub button1_click(sender as object, e as eventargs)
    labell1.text=convert.toString(convert.toInt32(Labell1.Text) + 1)
end sub
</script>
<html><body>
<form runat="server">
    <asp:button id="button1" text="Artır" onClick="button1_click"
        runat="server" /><br>
    <asp:label id="labell1" runat="server" />
</form></body></html>

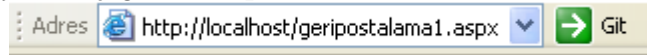
```



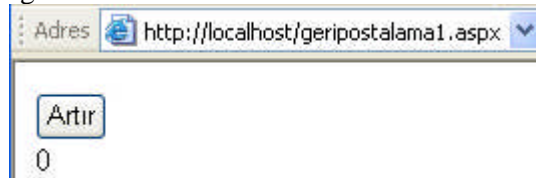
Sayfa ilk yüklendiğinde Düğmeye 3 defa tıklandığında
Resim 1.13: GeriPostalama1.aspx sayfasının tarayıcıdaki görüntüsü

Page_load olayında sayfa ilk yüklendiğinde **labell1**'in **text**'ine 0 (sıfır) değeri atanmıştır. **Page.ispostback** ifadesi sayfanın geri postalanma durumunu ifade eder. Kodlardaki **not page.ispostback** ifadesi de “sayfanın geri postalanmadığı durumu” ifade eder. 3-5.satırlardaki kodların anlamı şudur: “Eğer sayfa ilk defa yüklendiyse (geri postalanmadıysa) **labell1**'in **text**'ine 0 (sıfır) değerini ata”. **Button1_Click** alt programındaki 8.satırda ise **labell1**'in **Text** değeri artırılmaktadır. Sayfanın çalışması kısaca şu şekildedir:

- 1.Sayfa tarayıcıdan çağrılır. (**Request**)



- 2.Sunucuda sayfa yüklenir. (**Page_Load**)
- 3.Sayfanın ilk defa yüklendiği tespit edilir. (**if not page.ispostback then**)
- 4.**Label1**'in **text** değerine 0 değeri atanır. (**labell1.text="0"**)
- 5.Sayfa istemciye gönderilir.

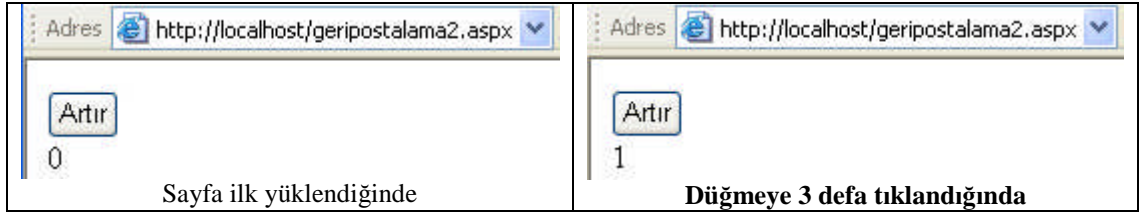


6. Düğmeye basılır (click). Sayfa istemciye gönderilir.
7. Sunucu sayfayı yükler. Sayfa ilk defa yüklenmediğinden 3-5. satırlar çalışmaz.
8. **Button1_click** olayı işlenir. **Label1**'in değeri bir artırılır.

label1.text=convert.toString(convert.toInt32(label1.text) + 1)

UYGULAMA

Sayfanın 3. ve 5. satırlarını silip **geriPostalama2.aspx** adıyla kaydedip, tarayıcıdan çağırınız. Sonucu yorumlayınız.



Resim 1.14: GeriPostalama2.aspx sayfasının tarayıcıdaki görüntüsü

1.11. Görünüm Durumu (ViewState)

ViewState (Görünüm Durumu), verileri saklamak için kullanılan yöntemlerden biridir. ViewState, kontrolü varsayılan olarak aktif durumdadır. ASP.NET, sayfa ve sayfadaki kontrollerle ilgili değerleri kodlanmış tek bir string hâlinde ViewState isimli bir gizli input kontrolünde tutar. Kaynak koda bakılarak bu kodlanmış veriler görülebilir. ViewState değerleri, istemci sayfasında tutulur, böylelikle sunucuda yer kaybına neden olmaz. Sayfa ve kontrollerle ilgili değerler arttıkça ViewState'in boyutu da artar. Bu durum da performans sıkıntısına neden olabilir. Ayrıca aspx sayfasının içinde bulunan bir kontrol olduğu için hem postback hem de request süresini uzatır. Gerektiğinde ViewState pasif hâle getirilebilir. Şimdi ViewState'in çalışmasıyla ilgili bir örnek yapalım. Örnekte bir DropDownList kontrolündeki ülke isimlerinin düğmeye basıldığında tersten sıralanması işlemi gerçekleştirilmektedir.

UYGULAMA

terstenSiralama.aspx

```
<script runat="server">
Sub Page_Load(Sender As Object, E As EventArgs)
    If Not Page.IsPostBack Then
        DropDownList1.Items.Add("Afganistan")
        DropDownList1.Items.Add("Almanya")
        DropDownList1.Items.Add("ABD")
        DropDownList1.Items.Add("Arjantin")
    End If
End Sub
Sub btnSiralama_Click(Sender As Object, E As EventArgs)
```

```

Dim ary as New ArrayList()
ary.AddRange(DropDownList1.Items)
ary.Reverse()
Dim obj As Object
DropDownList1.Items.Clear()
For Each obj in ary
    DropDownList1.Items.Add(obj.ToString())
Next
End Sub
</script>
<body>
<form ID="form1" runat="server">
    <asp:DropDownList ID="DropDownList1" runat="server"
        EnableViewState="True" />
    <asp:Button ID="Button1" runat="server" Text="Gönder"
        onClick="btnSiral_Click" />
</form></body></html>

```

Sayfanın çalışmasını inceleyelim:

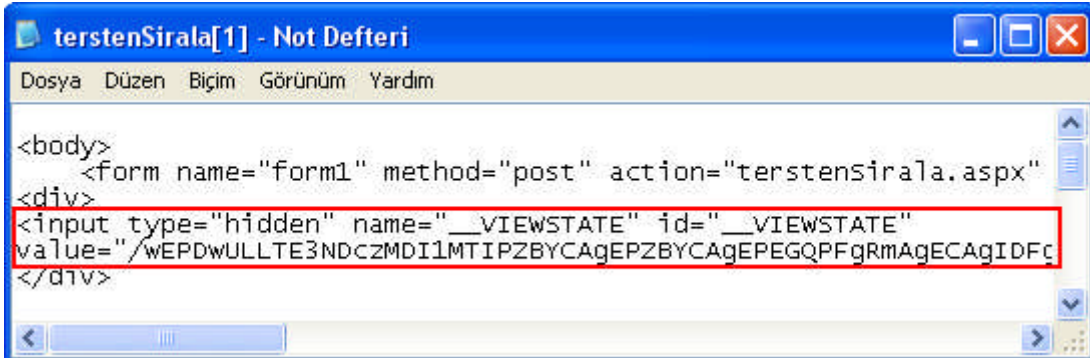
1. Sayfa ilk kez yüklendiğinde **DropDownList1**'e ülke isimleri eklenmiştir. (**DropDownList1.Items.Add("Afganistan")...**)



Resim 1.15: TerstenSiral.aspx sayfası ilk yüklendiğinde tarayıcıdaki görüntüsü

Bu durumda sayfanın kaynak koduna bakalım (Görünüm menüsü/Kaynak komutu). Sunucu, sayfaya ve sayfada bulunan kontrollerle ilgili bazı bilgileri gizli bir form elemanı olan **ViewState**'de tutarak sayfayı istemciye göndermiştir (Resim 1.16).

2. Gönder düğmesine tıklandığında sunucu btnSiral_Click alt programındaki kodlarla **DropDownList**'teki ülke isimlerini ters sıralamıştır ve sayfayı istemciye göndermiştir (Resim 1.17).



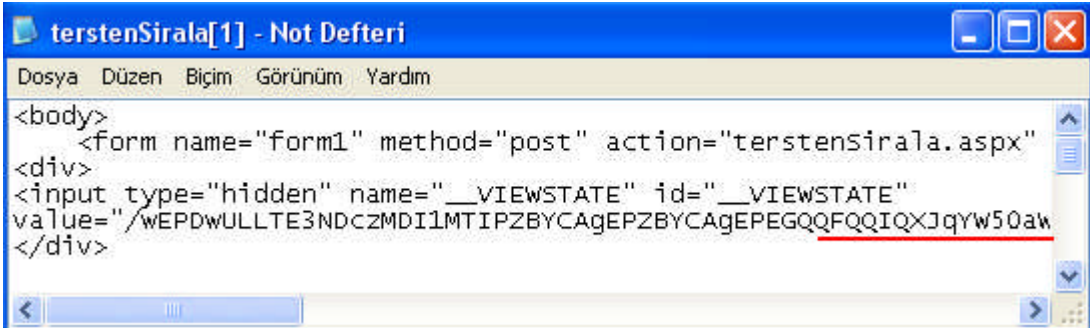
```
<body>
  <form name="form1" method="post" action="terstensiralas.aspx"
  <div>
    <input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
    value="/wEPDwULLTE3NDczMDI1MTIPZBYCAgEPZBYCAgEPEGQPFgRmAgECAgIDFc
  </div>
```

Resim 1.16: TerstenSiralas.aspx sayfasının kaynak kodları



Resim 1.17: Gönder düğmesine tıklandıktan sonra terstenSiralas.aspx'in tarayıcıdaki görüntüsü

Şimdi sayfanın kaynak koduna bakalım (Resim 1.18). Sunucu, sayfaya ve sayfada bulunan kontrollerle ilgili yeni duruma ait bilgileri **ViewState**'de tutarak sayfayı istemciye göndermiştir. Dikkat edilirse kaynak kodu görüntülerindeki **ViewState**'lerin içeriği birbirinden farklıdır.

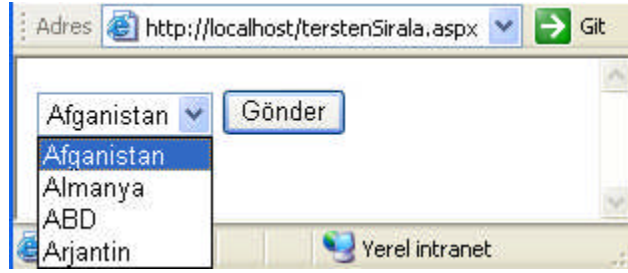


```
<body>
  <form name="form1" method="post" action="terstensiralas.aspx"
  <div>
    <input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
    value="/wEPDwULLTE3NDczMDI1MTIPZBYCAgEPZBYCAgEPEGQQIQXJqYw50aw
  </div>
```

Resim 1.18: Gönder düğmesine tıklandıktan sonra terstenSiralas.aspx'in kaynak kodları

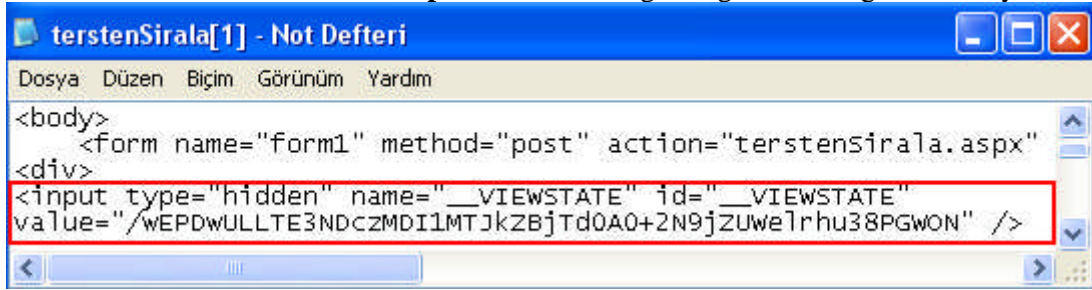
Şimdi de **DropDownList1**'in **ViewState**'ini kapatarak sayfayı inceleyelim. **DropDownList1**'in **EnableViewState** özelliğini **False** yapalım (**EnableViewState="False"**). Internet Explorer penceresini kapatıp tekrar açtıktan sonra sayfayı çağıralım.

NOT: Tarayıcıdan sayfa çağırıldığında bir oturum (session) açılır. Oturum kapatılmadığı sürece aynı oturuma devam edilir. Burada oturumu kapatıp sayfayı ilk defa çağırıyor olmak için tarayıcıyı kapatıp açıyoruz.



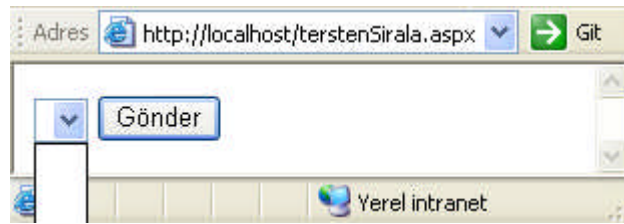
Resim 1.19: TerstenSiralas.aspx ilk yüklendiğinde tarayıcıdaki görüntüsü (Viewstate kapalı)

1. Sayfa ilk kez yüklendiğinde **DropDownList1**'e ülke isimleri eklenmiştir. Bu durumda sayfanın kaynak koduna bakıldığında **ViewState**'de **DropDownList1** ile ilgili bilgilerin tutulmadığı görülmektedir. **ViewState**'in **Value** değeri daha kısadır. Sunucuda da **DropDownList1** ile ilgili bilgi tutulmadığını unutmayalım.



Resim 1.20: Gönder düğmesi tıklandıktan sonra terstenSiralas.aspx'in kaynak kodları (Viewstate kapalı)

Gönder düğmesine basıldığında sayfa sunucuya ulaşmıştır. Sayfa ilk defa yüklenmediğinden **Page_Load** alt programında **if** bloğu çalışmamıştır. Bu nedenle **DropDownList1**'e herhangi bir değer eklenememiştir. Düğmeye tıklanmasıyla çalışan **BtnSiralas_Click** alt programında ise **DropDownList1**'in herhangi bir değeri olmadığından ters sıralama işlemi gerçekleşmemiştir.



Resim 1.21: Gönder düğmesine tıklandıktan sonra terstenSiralas.aspx'in tarayıcıdaki görüntüsü (Viewstate kapalı)

1.12. ASP.NET Web Sitesini Yapılandırma

ASP.NET'in birçok özelliği XML yapılandırma (**configuration**) dosyalarıyla özelleştirilebilir. Bu dosyalar .NET uygulamalarından okunabilirler. Bir ASP.NET sitesi, web site klasörünün içinde oluşturulan bir **web.config** dosyasıyla yapılandırılır. Web site klasörünün alt klasörlerine de web.config dosyaları ekleyerek daha ayrıntılı yapılandırma gerçekleştirilebilir.

Örneğin, web sitenizin belirli kısımlarına yöneticiler (**administrators**) dışındaki kullanıcıların erişimini engellemek istiyorsanız, web.config dosyanız aşağıdaki şekilde oluşturulabilir. Diğer bir web.config tarafından geçersiz kılınmadıkça bu web.config dosyasındaki yapılandırma geçerlidir.

```
<configuration>
  <system.web>
    <authorization>
      <allow roles="Admins"/>
      <deny users="*" />
    </authorization>
  </system.web>
</configuration>
```

Web.config dosyası uygulamadaki tüm sayfaların kullanabileceği ayarları tanımlar. Eğer herhangi bir sayfada bu dosyadaki ayardan farklı bir ayarlama yaparsanız bu ayar web.config dosyasındaki geçersiz kılar. Özetle uygulama düzeyinde ayarlamalar için web.config dosyasını kullanabilirsiniz. Bu ayarları sayfa düzeyindeki ayarlarla değiştirebilirsiniz.

1.13. Durum Yönetimi

İnternet ortamında istemci ile sunucu arasında sürekli bir bağlantının olmaması istemci ile sunucu arasındaki iletişimde bir dezavantaj oluşturur. Windows programlarında kullanılan tüm veriler bilgisayarda tutulmaktadır. Bu nedenle veriler üzerinde gerçekleştirilecek işlemler kolaylıkla yapılmaktadır. Global olarak tanımlanan değişkenler vasıtasıyla veriler programın genelinde kullanılmaktadır.

Net ortamında ise web form sunucuya postalandığında form verileri istemci bilgisayarda tutulmaz, sunucuya gönderilir, orada işlenir ve tekrar istemciye gönderilir. Yani veriler tek ortamda değildir, iki farklı ortam arasında (istemci ve sunucu) gidip gelmektedir. Bu dezavantajdan kurtulmak için ASP.NET'te durum yönetimi kavramı geliştirilmiştir. Çeşitli yöntemlerle verilerin istemci bilgisayarda ve sunucuda kalıcı olarak tutulması sağlanır. Bir anlamda durum yönetimi Windows programlarındaki global değişkenlerin görevlerini yerine getirilmektedir.

ASP.NET'te durum yönetimi teknikleri sunucu ve istemci taraflı durum yönetimi teknikleri olarak ikiye ayrılır:

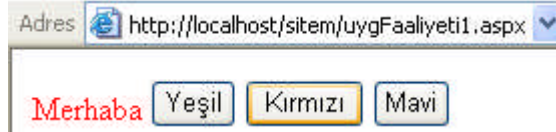
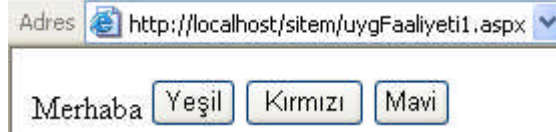
- İstemci Taraflı (Tabanlı) Durum Yönetimi Teknikleri
 - Görünüm Durumu (ViewState)
 - Gizli Form Alanları (Hidden Form Fields)
 - Çerezler (Cookies)
 - Sorgu Kelimeleri (Query Strings)

Sunucu Taraflı Durum Yönetimi Teknikleri:

- Oturum Durumu (Session State)
- Uygulama Durumu (Application State)
- Veritabanları (Databases)

UYGULAMA FAALİYETİ

Renk düğmelerine basıldığında label kontrolündeki yazının rengini değiştiren sayfayı oluşturunuz.



İşlem Basamakları	Öneriler
Uygun ad alanını ekleyiniz.	<code><%@ Import Namespace="System.Drawing" %></code>
Web form ekleyiniz.	<code><form runat="server" name="form1" method="post" action=""> </form></code>
Form içine 1 adet Label, 3 adet düğme kontrolü ekleyiniz.	<code><asp:Label ID="label1" runat="server" Text="Merhaba"> </asp:Label> <asp:Button ID="dugme1" runat="server" Text="Yeşil" OnClick="tiklandi" /></code>
<Script> etiketleri arasında tiklandi alt pogramını oluşturunuz.	<code>Sub tiklandi(Src As Object, E As EventArgs) End Sub</code>
Basılan düğmeyi tespit edip, uygun işlemi yapan kodları yazınız.	<code>Select Src.ID Case "dugme1" label1.ForeColor = Color.Green Case "dugme2"</code>

ÖLÇME VE DEĞERLENDİRME

Aşağıda verilen sorular için doğru cevap seçeneklerini işaretleyiniz.

1. Benzer görevleri yerine getiren sınıfların gruplandırılmasından oluşan yapıya ne denir?
A) Sınıf
B) Nesne
C) Ad alanı
D) Metot
2. Aşağıdakilerden hangisi ASP.NET sayfalarına otomatik olarak eklenen ad alanlarından biri değildir?
A) System
B) System.Web.UI.WebControls
C) System.Web.UI.HtmlControls
D) System.Drawings
3. Ad alanları programa hangi bildirim ile dâhil edilir?
A) @Register
B) @Import
C) @Page
D) @Control
4. Aşağıdakilerden hangisi doğrudur?
A) Textbox kontrolüne tıklanıldığında OnTextChanged olayı meydana gelir.
B) Kodun hata ayıklama modunda derlenmesi için Page bildirimde debug ifadesi kullanılır.
C) Olay gerçekleştiğinde işlenecek kodları içeren alt programa postback adı verilir.
D) Bir kontrolü diğer bir kontrole çevirmek için convert komutu kullanılır.
5. ASP.NET'te sayfanın ilk defa mı yüklendiğini yoksa geri mi postalandığını tespit etmek amacıyla hangi özellik kullanılır?
A) IsPostBack
B) AutoPostBack
C) Postback
D) GetPost

ÖĞRENME FAALİYETİ-2

AMAÇ

Web ortamı için kontrol nesnesi uygulamaları gerçekleştirebileceksiniz.

ARAŞTIRMA

Web form elemanlarının HTML form elemanlarıyla kıyaslandığında üstünlüklerinin neler olduğunu araştırınız.

2. WEB FORM ELEMANLARI

Öğrenme Faaliyeti 1’de web formunun ne olduğunu ve nasıl çalıştığını öğrendik. Bu öğrenme faaliyetinde ise web form elemanlarını (sunucu kontrolleri) Dreamweaver MX2004 (İngilizce) editörünü kullanarak inceleyeceğiz. Aslında şimdiye kadar yaptığımız tüm sayfalarda sunucu kontrollerinin bazılarını kullandık.

NOT: Uygun program kullanarak Dreamweaver çalışma alanı dilini Türkçeye çevirebilirsiniz.

Dreamweaver MX2004 ile ASP.NET uygulamaları oluşturmaya başlamadan önce gerekli birkaç program yüklemesi yapalım.

1. Dreamweaver’ı güncel tutmak amacıyla 7.0 sürümünü 7.0.1 sürümüne güncelleyiniz Güncelleme dosyasını aşağıda verilen net adresinden indirebilirsiniz. **www.adobe.com/support/dreamweaver/downloads_updaters.html#dwmx2004**.

Verilen İnternet sayfasında **Dreamweaver MX 2004 Updater** başlığı altında bulunan **English Windows Updater (20.1 MB)** linkini tıklayarak güncelleme dosyasını indiriniz. Dreamweaver açık değilken güncelleme dosyasını çalıştırınız.

2. Veritabanına bağlantı kurma işleminde ortaya çıkabilecek sorunun çözümü için download.macromedia.com/pub/dreamweaver/extensions/SP2DBFix1.0.2.mxp adresindeki uzantıyı (**extension**) yükleyiniz.

3.Web servisleriyle ilgili işlemleri yapmak için gerekli olan .NET SDK’yı aşağıda verilen net adresinden indirerek kurunuz. (.NET SDK kurulmadan önce .NET Redistributable paketi bilgisayara kurulmuş olmalıdır.)

msdn2.microsoft.com/en-us/netframework/aa731542.aspx.

Verilen İnternet sayfasında **.NET Framework Version 2.0 Software Development Kit** başlığı altındaki **Download x86 version** linkini tıklatınız. Gelen sayfada **Download** düğmesine tıklatarak dosyayı indiriniz.

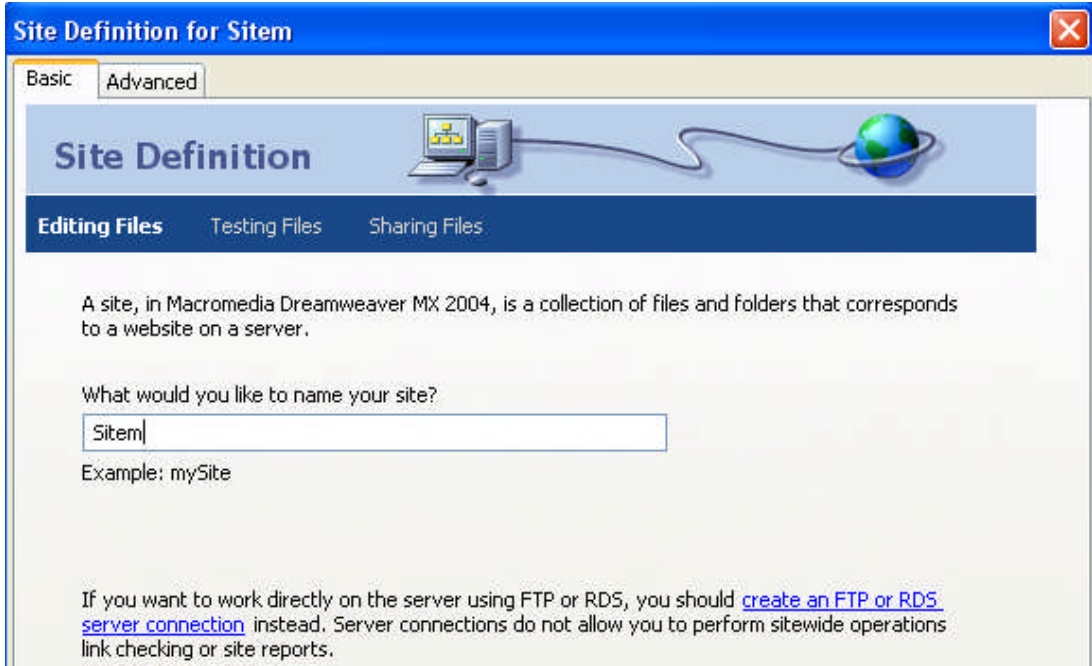
2.1. ASP.NET Sitesi Oluşturma

Öncelikle bir site tanımlayalım. **Dreamweaver Site...** seçeneğini tıklayarak site tanımlama (**Site Definition**) sihirbazını açınız (Resim 2.1).



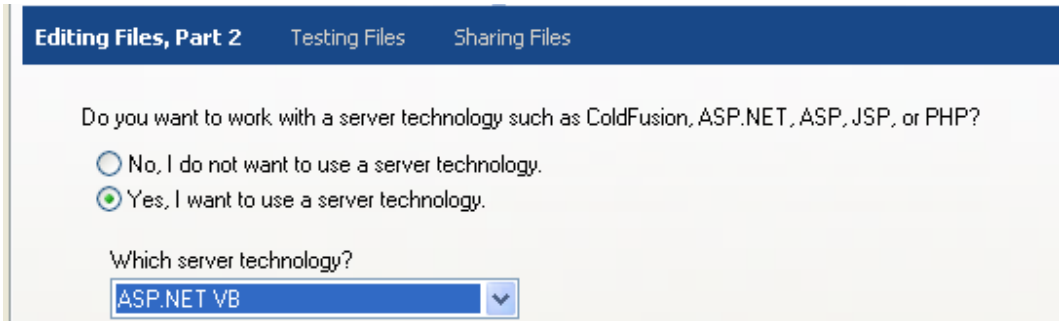
Resim 2.1: Dreamweaver başlangıç sayfası

Site adını yazmanızı isteyen soruya (**What would you like to name your site?**) Sitem yazarak **Next** düğmesini tıklatınız (Resim 2.2).



Resim 2.2: Site tanımlama

“ColdFusion, ASP.NET, ASP, JSP, PHP gibi bir sunucu teknolojiyle çalışmak ister misiniz?” (**Do you want to work with a server technology such as ColdFusion, ASP.NET, ASP, JSP, or PHP?**) sorusuna evet cevabını veriniz. Sunucu teknolojisi (**server technology**) olarak **ASP.NET VB**’yi seçiniz. **Next** düğmesini tıklatınız (Resim 2.3).



Resim 2.3: Sunucu teknolojisi belirleme

“Sitenizi geliştirirken dosyalarınızla nasıl çalışmak istersiniz?” (**How do you want to work with your files during development?**) sorusuna “Düzenle ve yerelde test et (Test sunucum bu makine üzerindedir)” [**Edit and test locally (my testing server is on this computer)**] cevabını veriniz. Böylelikle dosyalarınızı yerel bilgisayarınızda test edeceğinizi belirtmiş oldunuz. **Next** düğmesini tıklatınız (Resim 2.4).

Editing Files, Part 3 Testing Files Sharing Files


How do you want to work with your files during development?

Edit and test locally (my testing server is on this computer)

Edit locally, then upload to remote testing server

Edit directly on remote testing server using local network

Where on your computer do you want to store your files?



Because IIS has been installed on your computer, your computer can be used as a local testing server.

Resim 2.4: Dosyalarla nasıl çalışılacağını belirleme.

“Sitenizin kök klasörü için hangi URL’yi kullanmak istersiniz?” (**What URL would you use to browse to the root of your site?**) sorusuna karşılık varsayılan olarak gelen **http://localhost/sitem/** ifadesini değiştirmeden **Next** düğmesini tıklatınız (Resim 2.5).

“Bir dosyayı düzenlerken bu dosyayı diğer bir makineye kopyalar mısınız?” (**When you are done editing a file, do you copy it to another machine?**) sorusuna “Hayır” deyip **Next** düğmesini tıklatınız (Resim 2.6).

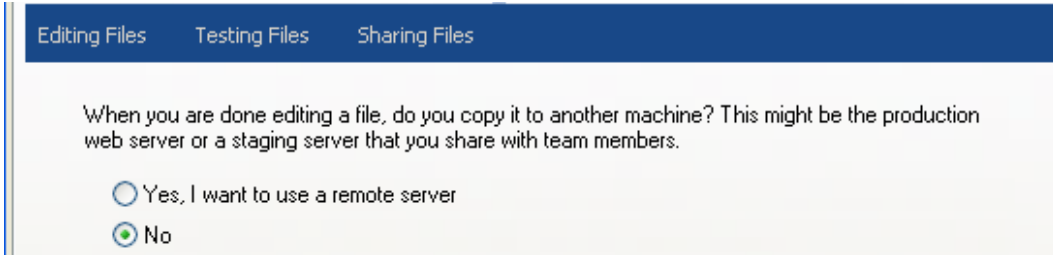
Editing Files **Testing Files** Sharing Files

Dreamweaver communicates with your testing server using HTTP (just like a browser), so it needs to know the URL of your site’s root folder.

What URL would you use to browse to the root of your site?

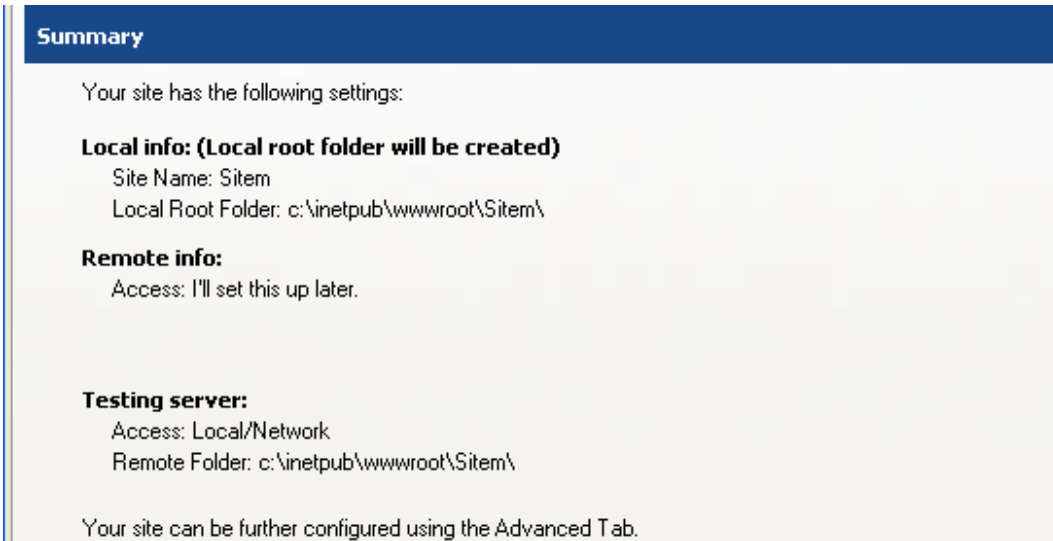
Example: http://ServerOne/RootFolder/

Resim 2.5: Kök klasörü için URL belirleme



Resim 2.6: Dosyayı paylaşma

Özeti (**Summary**) inceledikten sonra işlemi sonlandırmak için **Done** düğmesini tıklatınız (Resim 2.7).



Resim 2.7: Özet penceresi

Uygulama geliştirmeye başlamadan önce bazı dosyaları kurup işleme açmak (**deploy**) gereklidir. Sitenin kök klasöründe oluşturulan **bin** isimli klasöre destek dosyaları taşınmalıdır. Bin klasörü, uygulama için gerekli olan tüm montaj (assembly) yapısını barındırır.

Sitenizin kök klasöründe **bin** isimli bir klasör oluşturup **Site** menüsünden **Advanced** komutunu ardından **Deploy Supporting Files** komutunu veriniz. Bu komut **Deploy Supporting Files to Testing Server** penceresini açacaktır. Pencerede sorulan “Dreamweaver, sunucu davranışlarında kullanılan özel etiketlerin ve web servislerinde kullanılan temsilcilerin (**Proxy**) kurulup işleme açılmasına ihtiyaç duyar. Bin klasörüne nasıl erişeceksiniz?” sorusuna karşılık aşağıdaki düzenlemeleri yapınız. Ardından **Deploy** düğmesini tıklatınız (Resim 2.8).



Resim 2.8: Bin klasörüne erişimi belirleme

DreamweaverCtrls.dll dosyası **bin** klasörü içinde oluşturulduğunu söyleyen pencerede **Tamam** düğmesini tıklatınız (Resim 2.9).

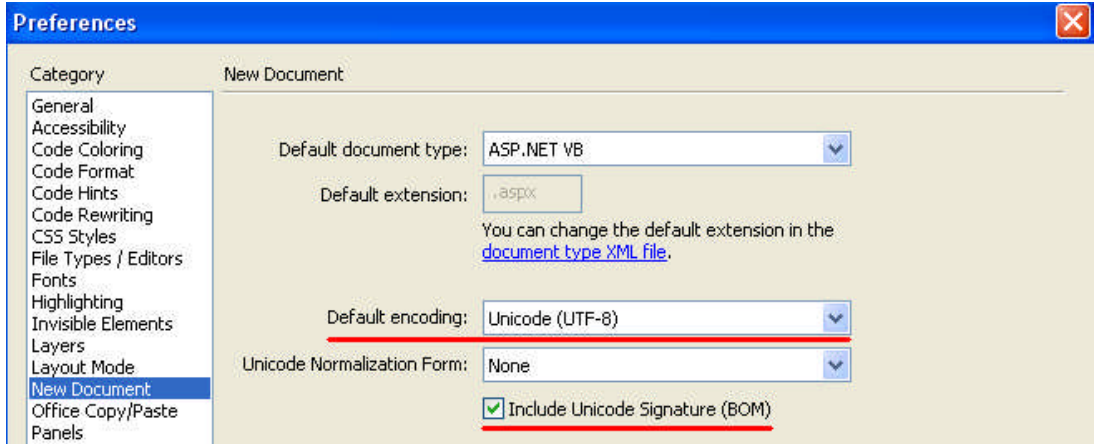


Resim 2.9: DreamweaverCtrls.dll dosyasının oluşturulduğunu belirten mesaj

Yeni **ASP.NET VB** sayfası oluşturmadan önce dil kodlamasını ayarlayalım. **Edit** menüsünden **Preferences...** komutunu tıklatıp gelen penceredeki **Category** listesinden **New Document** kategorisini seçiniz. Burada **Default Encoding** olarak **Unicode (UTF-8)**'i seçip **Include Unicode Signature (BOM)**'u işaretleyiniz. **Ok** düğmesini tıklatınız (Resim 2.10).

Yeni bir sayfası oluşturmak için **ASP.NET VB** seçeneğini seçiniz (Resim 2.11).

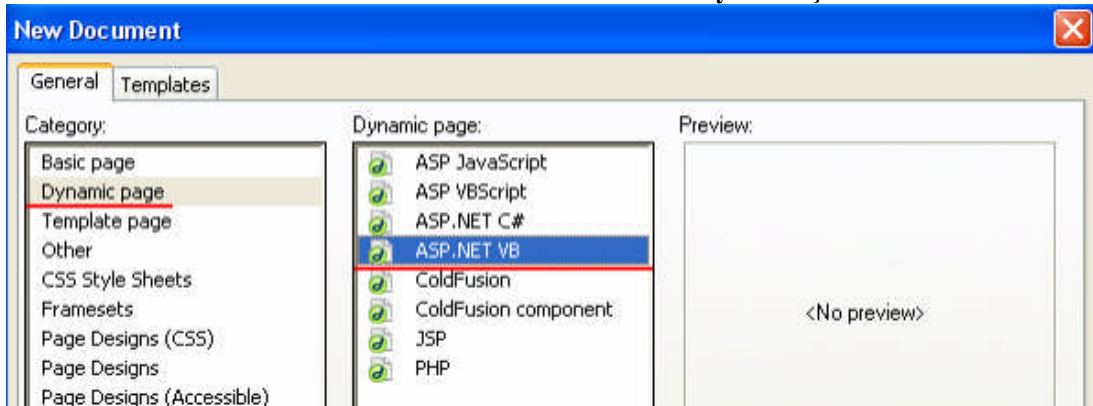
Yeni **ASP.NET VB** sayfası oluşturma işlemini başlangıç penceresinde **More...** seçeneğine tıklatınca gelen **New Document** penceresinden de gerçekleştirebiliriz. Bu pencereden önce **dynamic page**, ardından **ASP.NET VB** seçenekleri tıklanmalıdır. Belgenizin XHTML uyumlu olmasını isterseniz **Make document XHTML compliant** onay kutusunu işaretleyebilirsiniz (Resim 2.12).



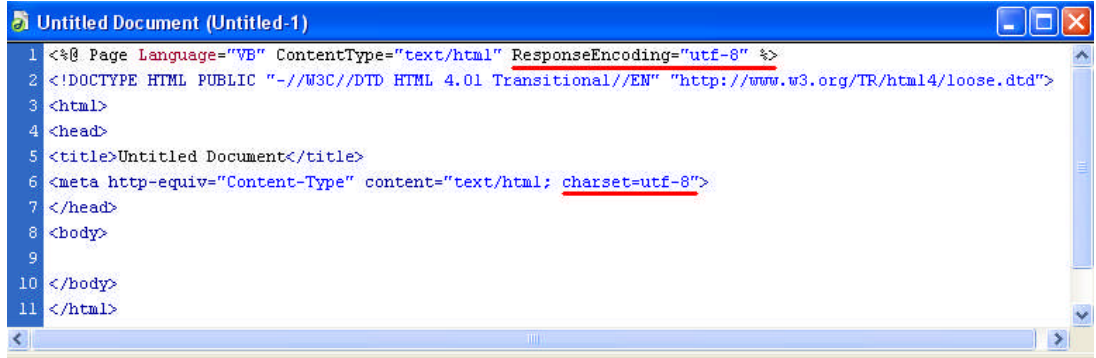
Resim 2.10: Dil kodlamasını ayarlama



Resim 2.11: Yeni bir ASP.NET VB sayfası oluşturma



Resim 2.12: New Document penceresinden yeni bir ASP.NET VB sayfası oluşturma



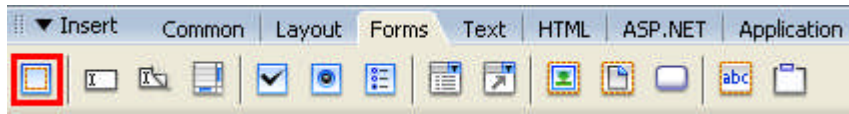
```
1 <%@ Page Language="VB" ContentType="text/html" ResponseEncoding="utf-8" %>
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
3 <html>
4 <head>
5 <title>Untitled Document</title>
6 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
7 </head>
8 <body>
9
10 </body>
11 </html>
```

<body> 772 x 53 1K / 1 sec

Resim 2.13: Oluşturulan ASP.NET VB sayfası

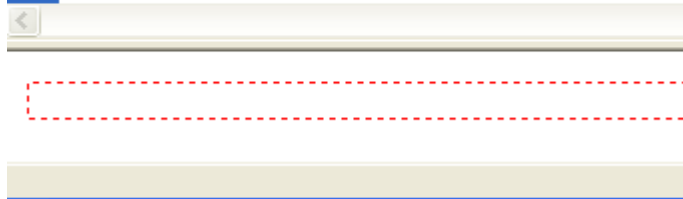
2.2. Web Form Oluşturma

Insert çubuğundan **Forms** sekmesini (tab) tıklatınız. (Insert çubuğu **Show as Tabs** görünümündedir.) **Design view** (tasarım görünümü) penceresinde web formuna tıklatıp ardından **Form** simgesini tıklatınız (Resim 2.14).



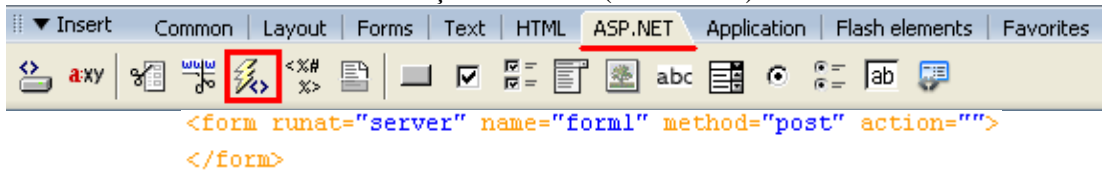
Resim 2.14: Sayfaya form ekleme

```
8 <body>
9 <form name="form1" method="post" action="">
10 </form>
11 </body>
```



Resim 2.15: Eklenen formun kod ve tasarım görünümü

Kod penceresinde form elemanının **name** özelliğinin önüne tıklatınız. **Insert** çubuğunda **ASP.NET** sekmesine geçip, **Runat Server** simgesine tıklatınız. Eklenen **runat="server"** kodundan sonra boşluk bırakınız (Resim 2.16).



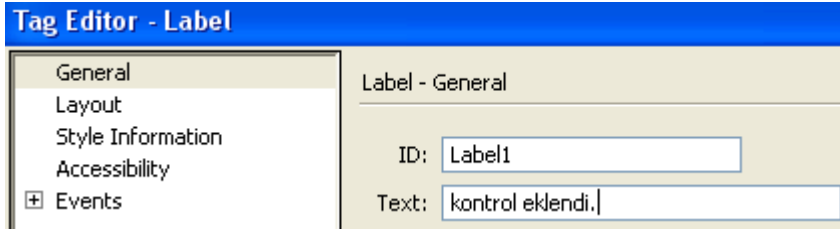
Resim 2.16: Formun web form hâline dönüştürülmesi

İşte Dreamweaver kullanarak ilk web formunuzu oluşturduunuz. **Code View** penceresinde imleç **<body>** etiketleri arasındayken form simgesine tıklatarak veya sürükleyip bırak yöntemiyle de yeni form eklenebilir.

2.3. Web Forma Kontrol Ekleme

Code View'de form etiketleri arasında kontrolü ekleyeceğimiz satırı açtıktan sonra **ASP.NET** sekmesindeki **asp:Label** simgesine tıklatınız. **Tag Editor – Label** penceresinde **ID** ve **Text** değerlerini girdikten sonra **OK** düğmesini tıklatınız (Resim 2.17).

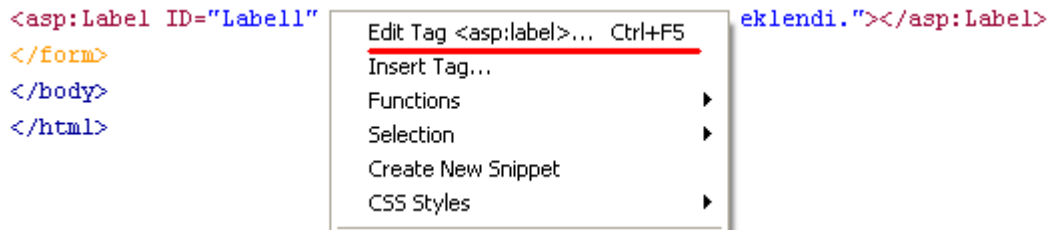
```
<form runat="server" name="form1" method="post" action="">
</form>
```



Resim 2.17: Kontrolün özelliklerini belirleme

```
<form runat="server" name="form1" method="post" action="">
<asp:Label ID="Label1" runat="server" Text="kontrol eklendi."/></asp:Label>
</form>
```

Kontrolün özelliklerini değiştirmek için **Code View** ve **Design View**'de kontrolün üzerinde kısayol menüsünü getirerek **Edit Tag...** komutunu kullanabilirsiniz (Resim 2.18).



Resim 2.18: Tag Editor penceresini açma

2.4. Sayfayı Kaydetme

Sayfayı kaydetmek için **File** menüsünden **Save** komutunu veriniz. **Save As** penceresinde **Kayıt Türü** kısmından **Active Server Plus Page (*.aspx;*.ascx;*.asmx...)** seçeneğini seçiniz. **Dosya adı** kısmındaki **Untitled-2** geçici ismi otomatik olarak **Untitled-2.aspx** olarak değişecektir. Son adım sayfa içeriğine uygun bir isim verdikten sonra **Kaydet** düğmesini tıklamaktır.

2.5. Web Form Elemanları (Sunucu Kontrolleri)

Sunucu kontrollerinin tümü, sunucudadır. Sunucu kontrolleri nesne olarak değerlendirilir. Web uygulama geliştiricisi uygulamasını bu nesnelere geliştirir. Sunucu, uygulamayı istemci bilgisayara HTML formatıyla gönderir.

Sunucu kontrolleri, özelliklere (properties), metotlara (method) ve olaylara (events) sahip olduğundan programlanabilir. Programcı, ASP.NET ile birlikte gelen sunucu kontrollerinden birini kullanabildiği gibi oluşturduğu kendi kontrollerini de kullanabilir. Bu kontrolleri **HTML sunucu kontrolleri** ve **web sunucu kontrolleri** olarak ikiye ayırabiliriz.

2.6. HTML Sunucu Kontrolleri

System.Web.UI.HtmlControls ad alanındaki sınıflara ait sunucu kontrolleridir. Web kontrolleriyle HTML kontrolleri arasında görüntü açısından fark olmamakla birlikte web sunucu kontrolleri daha fazla özelliğe sahiptir. Peki daha işlevsel olan web kontrolleri varken neden HTML kontrolleri kullanılır? Bunun sebebi, daha önce oluşturulan web sayfalarını ASP.NET sayfalarına çevirmektir. Tabii ki uygulama geliştiricisi uygulamasını en baştan ASP.NET'te yapmayı tercih edebilir. Eğer web uygulamamızı yeni oluşturuyorsak web sunucu kontrollerini kullanmak daha yararlıdır.

HTML sunucu kontrollerini kullanmak için **System.Web.UI.HtmlControls** ad alanını import bildiriyle eklemeye gerek yoktur. Çünkü bu ad alanı ASP.NET tarafından otomatik olarak sayfaya eklenir.

HTML elemanını HTML sunucu kontrolüne çevirmek için o elemanın bildirimine `runat="server"` ifadesi eklenmelidir. Ayrıca, o kontrolü diğer kontrollerden ayırmayı sağlayan **id** bildiri de kullanılır. Örneğin, bir HTML form elemanı olan butonu sunucu kontrolüne çevirelim.

```
<input type = "submit" id="buton1" runat="server">
```

Artık, bu buton HTML sunucu kontrolü olmuştur. **HTMLButon** sınıfı kullanılarak kontrolün yönetimi gerçekleştirilecek ve HTML butonu, bir nesne olarak özellikleri, metotları ve olayları kullanılacaktır. HTML elemanlarını HTML sunucu kontrollerine dönüştürerek **HTMLOrnek.aspx** sayfasını oluşturalım:

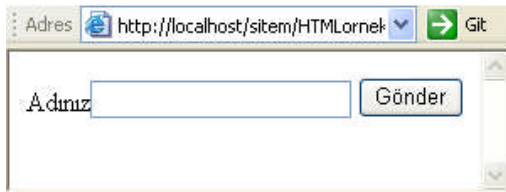
- Yeni bir sayfa oluşturunuz.
- Sayfaya yeni form ekleyip ardından `runat=server` ifadesini ekleyerek formu web forma dönüştürünüz.
- `<body>` etiketleri arasına "Adınız" yazınız.
- Forms çubuğundan forma **Text Field** ekleyiniz. **General** kategorisinde **Type** listesinden **text**'i, **Style Sheet/Accessibility** kategorisinde **ID**'yi **text1** olarak belirleyiniz. **Runat=server** ifadesini ekleyiniz.

- Forms çubuğundan forma **Text Field** ekleyiniz. **General** kategorisinde **Type** listesinden **Submit**'i seçip, **Value** kutusuna **Gönder** metnini yazınız. **Style Sheet/Accessibility** kategorisinde **ID**'yi **button1** olarak belirleyiniz. Bu etiketin arasına **Runat=server** ve **onServerClick="goruntule"** ifadelerini ekleyiniz.
- <head> etiketleri arasına aşağıdaki kod satırlarını yazınız.

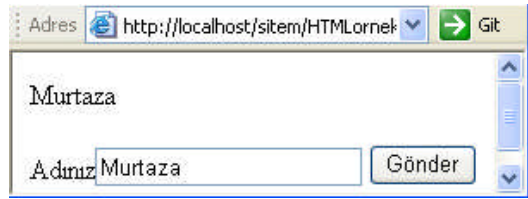
```
<script runat="server">
sub goruntule(sender as object, e as eventargs)
    response.write(text1.value)
end sub
</script>
```

HTMLornek.aspx

```
<script runat="server">
sub goruntule(sender as object, e as eventargs)
    response.write(text1.value)
end sub
</script>
</head>
<body>
<form runat="server" name="form1" method="post" action="">
Adınız<input runat="server" name="" type="text" id="text1">
<input name="" type="submit" id="button1" value="Gönder"
runat="server" onServerClick="goruntule" >
</form>
```



Resim 2.19: Htmllornek sayfası



Resim 2.20: Form geri postalandıktan sonra

Sayfamızda daha önce **onClick** olarak görmeye alışık olduğunuz ifadenin **onServerClick** şeklinde yazıldığını fark ettiniz mi? Bu farklılığın sebebi, HTML form elemanlarının istemci taraflı kodlama olaylarının tanımlanma ihtimalidir. Yani, **onClick** ifadesi kullanıldığında bu ifadenin, olayı istemci taraflı mı yoksa sunucu taraflı mı ifade edeceği konusunda karışıklık ortaya çıkacaktır. İşte, bu karışıklığı engellemek ve olayın sunucu olayı olduğunu belirtmek için **server** kelimesini kullanırız.

UYGULAMA:

- Onserverclick ifadesi yerine onclick yazarak sayfayı kaydediniz. Ardından sayfayı çağırarak adınızı yazıp Gönder düğmesine basınız. Sonucu yorumlayınız.

- b) Sayfanın ilk hâlinde <input....> etiketindeki runat="server" ifadesini silip sayfayı kaydediniz. Ardından sayfayı çağırarak adınızı yazıp Gönder düğmesine basınız. Sonucu yorumlayınız.

Aşağıdaki tabloda bazı HTML etiketleri, bu etiketleri HTML sunucu kontrolü olarak kullanmamızı sağlayan System.Web.UI.HtmlControls ad alanındaki ilgili sınıf ve sınıfın amacı yer almaktadır.

HTML etiketi	İlgili sınıf	Amacı
<a>	HtmlAnchor	Html <a> etiketine program yoluyla ulaşmayı sağlar.
<button>	HtmlButton	Html <button> etiketine program yoluyla ulaşmayı sağlar.
<form>	HtmlForm	Html <form> etiketine program yoluyla ulaşmayı sağlar.
	HtmlImage	Html etiketine program yoluyla ulaşmayı sağlar.
<input type="button"> <input type="submit"> <input type="reset">	HtmlInputButton	Html input etiketinin button, submit, reset türlerine program yoluyla ulaşmayı sağlar.
<input type="checkbox">	HtmlInputCheckBox	Html input etiketinin checkbox türüne program yoluyla ulaşmayı sağlar.
<input type="image">	HtmlInputImage	Html input etiketinin image türüne program yoluyla ulaşmayı sağlar.
<input type="radio">	HtmlInputRadioButton	Html input etiketinin radio türüne program yoluyla ulaşmayı sağlar.
<input type="text"> <input type="password">	HtmlInputText	Html input etiketinin text ve password türlerine program yoluyla ulaşmayı sağlar.
<select>	HtmlSelect	Html <Select> etiketine program yoluyla ulaşmayı sağlar.
<table>	HtmlTable	Html <Table> etiketine program yoluyla ulaşmayı sağlar.
<td> ve <th>	HtmlTableCell	Html tablo hücresi öğelerine program yoluyla ulaşmayı sağlar.
<tr>	HtmlTableRow	Html tablo satırı öğelerine program yoluyla ulaşmayı sağlar.
<textarea>	HtmlTextArea	Html <textarea> (metin alanı) öğesine program yoluyla ulaşmayı sağlar.
<body>, <div>, , vs.	HtmlGenericControl	Kendi Html sınıfı tarafından özel olarak temsil olunmayan her Html öğesine program yoluyla ulaşmayı sağlar.

Tablo 2.1: HTML etiketleri, ilişkili .NET sınıfı, sınıfın amacı

Tablodan anlaşılacağı üzere HTML elemanlarının .NET'te bir sınıf karşılığı vardır. Her sınıfın kendine ait özellikleri, metotları, olayları vardır. Gerektiğinde diğer HTML elemanlarıyla ilgili bilgiyi .NET SDK dokümanlarından, Microsoft'un MSDN sayfalarından ve ilgili net sitelerinden öğrenebilirsiniz.

2.7. Web Sunucu Kontrolleri

System.Web.UI.WebControls ad alanında tanımlı sınıflardan oluşturulan kontrollerdir. HTML sunucu kontrollerinden daha fazla özelliğe sahip olduklarından programlanabilme açısından daha uygundur. Web programcısı, var olan web sunucu kontrollerini kullanabildiği gibi kendi ihtiyacına göre geliştirdiği kontrolleri de kullanabilir. Web sunucu kontrollerinin bir kısmını zaten daha önce kullanmıştık. Bu bölümde sık kullanılan diğer kontrolleri inceleyeceğiz.

Web sunucu kontrollerini kullanmak için System.Web.UI.WebControls ad alanını import bildirimisiyle eklemeye gerek yoktur. Çünkü bu ad alanı ASP.NET tarafından otomatik olarak sayfaya eklenir. Web sunucu kontrollerinin altyapısını oluşturan System.Web.UI.WebControls ad alanındaki sınıfların bir kısmı aşağıda belirtilmiştir;

AdRotator	Button	Calendar	CheckBox	CheckBoxList
DropDownList	HyperLink	Image	ImageButton	Label
LinkButton	ListBox	ListControl	ListItem	Literal
Panel	PlaceHolder	RadioButton	RadioButtonList	Table
TextBox	Xml			

Tablo 2.2: System.Web.UI.WebControls ad alanındaki bazı sınıflar

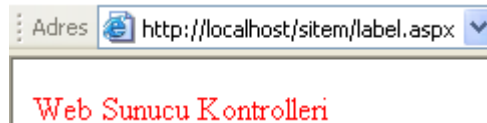
Web sunucu kontrolleri sadece form elemanlarını değil, özel amaçlı kontrolleri de içermektedir. Örneğin, Calendar (takvim) nesnesi.

2.7.1. Label

Web sayfası kullanıcılarına mesaj iletmek için kullanılır. Kullanıcı tarafından label'in metni değiştirilemez.

label.aspx

```
<form name="form1" method="post" action="">
<asp:Label ForeColor="#FF0000" ID="label1" runat="server" _
Text="Web Sunucu Kontrolleri" EnableViewState="false"></asp:Label>
</form>
```



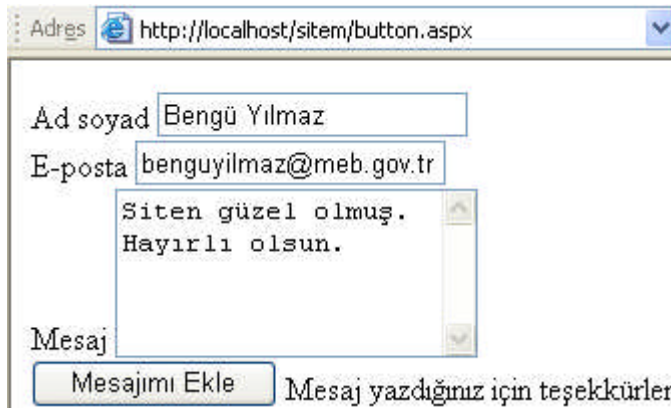
Resim 2.21: Label.aspx sayfasının tarayıcıdaki görüntüsü

Label kontrolünün değeri sunucuda işlenmeyeceğinden **EnableViewState**'e **false** değeri verilmiştir.

2.7.2. TextBox

Kullanıcının metin girmesini sağlayan kontroldür.
mesajYaz.aspx

```
<script language="VB" runat="server">
sub ekle(sender as object, e as eventargs)
lbltesekkur.text = "Mesaj yazdığınız için teşekkürler."
end sub
</script>
</head>
<body>
<form runat="server" name="form1" method="post" action="">
<asp:Label ID="lbladsoyad" runat="server" Text="Ad soyad"></asp:Label>
<asp:TextBox ID="txtadsoyad" runat="server" /><br>
<asp:Label ID="lbleposta" runat="server" Text="E-posta"></asp:Label>
<asp:TextBox ID="txteposta" runat="server" /><br>
<asp:Label ID="lblmesaj" runat="server" Text="Mesaj"></asp:Label>
<asp:TextBox ID="txtmesaj" Rows="5" runat="server"
TextMode="MultiLine" /><br>
<asp:Button ID="btnekle" runat="server" Text="Mesajımı Ekle"
OnClick="ekle" />
<asp:Label ID="lbltesekkur" runat="server"></asp:Label>
</form>
```



Resim 2.22: Button.aspx sayfasının tarayıcıdaki görüntüsü

Label kontrollerinin id özellikleri belirlenirken label'i temsilen **lbl** harfi (lblad, lbleposta, lblmesaj), textbox kontrollerinin id özellikleri belirlenirken **txt** harfleri (txtad, txteposta, txtmesaj) ismin başında kullanılmıştır. Sayfada mesajın yazılacağı metin kutusunun birden fazla satır olacağı **textmode="multiline"** ifadesiyle, beş satır olacağı **rows="5"** ifadesiyle belirlenmiştir.

Uygulama

MesajYaz.aspx sayfasında hiçbir bilgiyi doldurmadan düğmeye basınız. Sonucu yorumlayınız.

2.7.3. Button

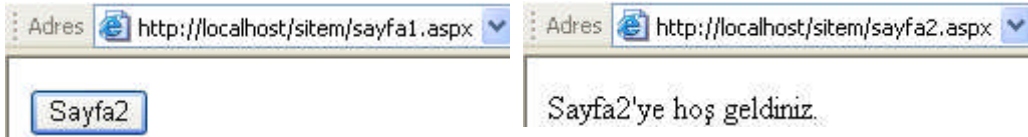
Düğme eklemek için kullanılır. Sayfa1.aspx sayfasındaki düğmeye tıkladığında Sayfa2.aspx sayfasına yönlendirme yapan bir örnek sayfa oluşturalım.

sayfa1.aspx

```
<script runat="server">
Sub ikinciSayfa(Src As Object, E As EventArgs)
response.Redirect("sayfa2.aspx")
End Sub
</script></head>
<body>
<form runat="server" name="form1" method="post" action="">
<asp:Button ID="button1" runat="server" Text="Sayfa2" OnClick="ikinciSayfa" />
</form>
```

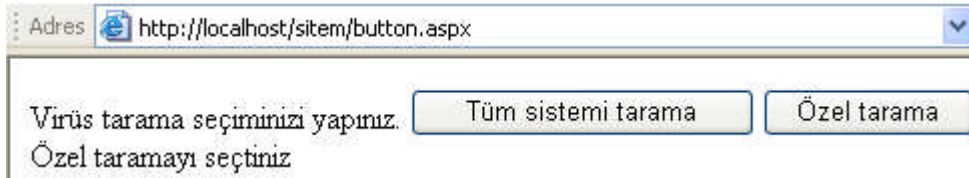
sayfa2.aspx

```
<body>
Sayfa2'ye hoş geldiniz.
</body>
```



Resim 2.23: Sayfa1.aspx ve sayfa2.aspx sayfalarının tarayıcıdaki görüntüsü

```
<script runat="server">
sub secim(nesne as object, e as EventArgs)
lblmesaj.text = nesne.text & "yı seçtiniz"
end sub
</script>
</head>
<body>
<form runat="server" action="" method="post">
Virüs tarama seçiminizi yapınız.
<asp:Button ID="btntum" runat="server" Text="Tüm sistemi tarama" OnClick="secim" />
<asp:Button ID="btnoz" runat="server" Text="Özel tarama" OnClick="secim" /><br>
<asp:Label ID="lblmesaj" runat="server"></asp:Label>
</form>
```



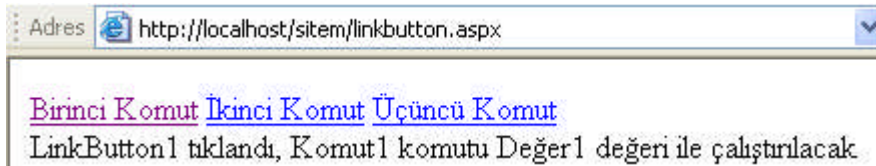
Resim 2.24: Tarama.aspx sayfasının tarayıcıdaki görüntüsü

2.7.4. LinkButton

Bir alt programı çağıran tıklanabilir yazı linki (bağlantı, köprü) görüntüler. Buton kontrolünün metin karşılığıdır. Buton kontrolü gibi web form sayfasını sunucuya postalar. Hyperlink kontrolüne benzer fakat buton kontrolü ile aynı işlevlere sahiptir. Aynı göreve sahip Button ve LinkButton kontrollerinin seçimi geliştiricinin görsel tercihinine bağlıdır. **OnClick** olayı kullanılarak kontrole tıklandığında çalıştırılması istenen kodlar yazılabilir. İstenirse **Response.Redirect(URL)** komutu ile istenilen sayfaya yönlendirme yapılır. **PostBackUrl** ile bulunulan sayfadan farklı bir sayfaya postalama işlemi yapılır. **Insert** çubuğunun **ASP.NET** sekmesindeki **More Tags...** simgesine tıklanınca gelen **Tag Chooser** penceresindeki **ASP.NET Tags** bölümünden **LinkButton** ekleyebilirsiniz.

linkbutton.aspx

```
<script runat="server">
Sub KomutCalistir(Src As Object, E As CommandEventArgs)
Label1.Text = Src.ID & " tıklandı, " & E.CommandName & " komutu " & _
E.CommandArgument & " değeri ile çalıştırılacak"
End Sub
</script></head>
<body>
<form runat="server" name="form1" method="post" action="">
<asp:LinkButton CommandName="Komut1" CommandArgument="Değer1" ID="LinkButton1"
runat="server" Text="Birinci Komut" OnCommand="KomutCalistir"/>
<asp:LinkButton CommandName="Komut2" CommandArgument="Değer2" ID="LinkButton2"
runat="server" Text="İkinci Komut" OnCommand="KomutCalistir"/>
<asp:LinkButton CommandName="Komut3" CommandArgument="Değer3" ID="LinkButton3"
runat="server" Text="Üçüncü Komut" OnCommand="KomutCalistir"/><br>
<asp:Label ID="Label1" runat="server"></asp:Label>
</form>
```



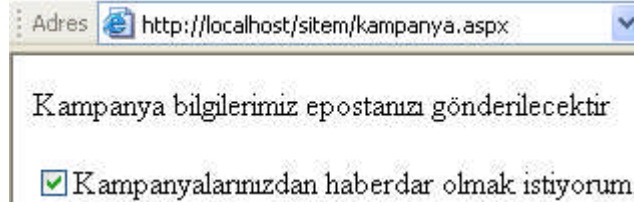
Resim 2.25: LinkButton.aspx sayfasının tarayıcıdaki görüntüsü

2.7.5. CheckBox

CheckBox, kullanıcının seçebileceği onay kutusu oluşturur.

kampanya.aspx

```
<script runat="server">
sub secim(sender As Object, e As System.EventArgs)
response.write("Kampanya bilgilerimiz epostanıza gönderilecektir")
end sub
</script>
</head>
<body>
<form runat="server" name="form1" method="post" action="">
<asp:CheckBox AutoPostBack="true" ID="checkboxox1" runat="server"
Text="Kampanyalarınızdan haberdar olmak istiyorum."
OnCheckedChanged="secim" />
</form>
```



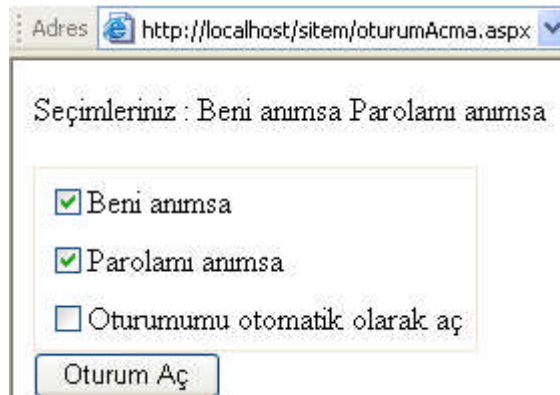
Resim 2.26: Kampanya.aspx sayfasının tarayıcıdaki görüntüsü

2.7.6. CheckBoxList

CheckBoxList, bir grup onay kutusu oluşturur.

oturumAcma.aspx

```
<script runat="server">
sub gonder(sender as object, e as EventArgs)
    dim secimler as string
    secimler="Seçimleriniz : "
    if secenekler.Items(0).Selected then secimler = _
    secimler + " " + secenekler.Items(0).text
    if secenekler.Items(1).Selected then secimler = _
    secimler + " " + secenekler.Items(1).text
    if secenekler.Items(2).Selected then secimler = _
    secimler + " " + secenekler.Items(2).text
    response.write(secimler)
end sub
</script></head>
<body>
<form runat="server" action="" method="post">
<asp:CheckBoxList BorderWidth="1" CellPadding="3" CellSpacing="3"
ID="secenekler" RepeatLayout="table" runat="server">
    <asp:ListItem>Beni anımsa</asp:ListItem>
    <asp:ListItem>Parolamı anımsa</asp:ListItem>
    <asp:ListItem>Oturumumu otomatik olarak aç</asp:ListItem>
</asp:CheckBoxList>
<asp:Button ID="btngonder" runat="server" Text="Oturum Aç"
OnClick="gonder" />
</form>
```



Resim 2.27: OturumAcma.aspx sayfasının tarayıcıdaki görüntüsü

Uygulama

Aşağıda tarayıcı görüntüsü verilen ASP.NET sayfasını geliştiriniz.



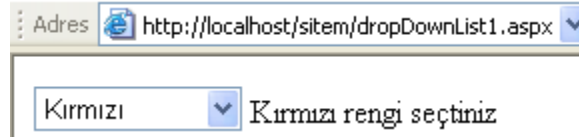
Resim 2.28: Kitaplar.aspx sayfasının tarayıcıdaki görüntüsü

2.7.7. DropDownList

Aşağı açılan bir liste görüntüleyen kontroldür. Çeşitli veri kaynaklarındaki veriler bu kontrole bağlanabilir.

dropDownList1.aspx

```
<script runat="server">
sub secim(sender as object, e as EventArgs)
    if renkler.selectedvalue="Renkler"
        labell.text = "Renk seçmediniz"
    else
        labell.text = renkler.selectedvalue & " rengi seçtiniz"
    end if
end sub
</script>
</head>
<body>
<form runat="server" name="form1" method="post" action="">
<asp:DropDownList AutoPostBack="true" ID="renkler" _
runat="server" OnSelectedIndexChanged="secim">
    <asp:ListItem Text="Renk Seçiniz" Value="Renkler" Selected="true" />
    <asp:ListItem Text="Kırmızı" Value="Kırmızı" />
    <asp:ListItem Text="Mavi" Value="Mavi" />
</asp:DropDownList>
<asp:Label ID="labell" runat="server"></asp:Label>
</form>
```

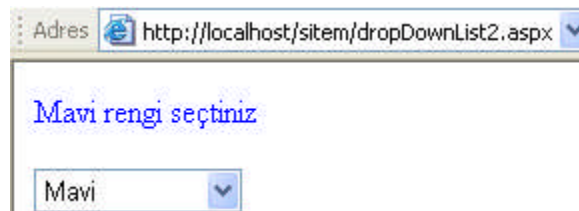


Resim 2.29: DropDownList1.aspx sayfasının tarayıcıdaki görüntüsü

dropDownList2.aspx

```
<script runat="server">
sub secim(sender as object, e as EventArgs)
    Select case renkler.selectedvalue
        case "Kırmızı"
            response.write("<font color='" & "Red" & "'>")
            response.write("Kırmızı rengi seçtiniz")
            response.write("</font>")
        case "Mavi"
            response.write("<font color='" & "Blue" & "'>")
            response.write("Mavi rengi seçtiniz")
            response.write("</font>")
        case "Yeşil"
            response.write("<font color='" & "Green" & "'>")
            response.write("Yeşil rengi seçtiniz")
            response.write("</font>")
    End Select
end sub
</script>

<form runat="server" name="form1" method="post" action="">
<asp:DropDownList AutoPostBack="true" ID="renkler"
runat="server" OnSelectedIndexChanged="secim">
    <asp:ListItem>Renk Seçiniz</asp:ListItem>
    <asp:ListItem>Kırmızı</asp:ListItem>
    <asp:ListItem>Mavi</asp:ListItem>
    <asp:ListItem>Yeşil</asp:ListItem>
</asp:DropDownList>
</form>
```



Resim 2.30: DropDownList2.aspx sayfasının tarayıcıdaki görüntüsü

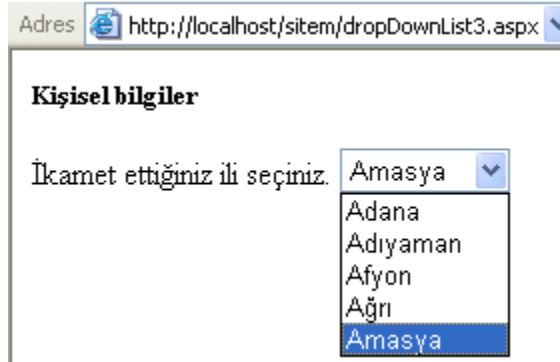
Uygulama

DropDownList'ten "Renk Seçiniz" seçeneğini seçiniz. Seçim sonrasında tarayıcıdaki görüntüyü yorumlayınız.

Verilerin DropDownList'e bağlanarak (**data binding**) görüntülediği bir örnek yapalım.

dropDownList3.aspx

```
<script runat="server">
dim baslik as string = "Kişisel bilgiler"
dim mesaj as string = "İkamet ettiğiniz ili seçiniz."
dim iller() as string = {"Adana","Adıyaman","Afyon","Ağrı","Amasya"}
Sub Page_Load(Src As Object, E As EventArgs)
page.DataBind
End Sub
</script></head>
<body>
<form runat="server" name="form1" method="post" action="">
<h5><%=baslik%></h5>
<asp:Label ID="labell" runat="server" Text="<%= mesaj %>"></asp:Label>
<asp:DropDownList DataSource="<%= iller %>" ID="liste" runat="server">
</asp:DropDownList>
</form>
```



Resim 2.31: DropDownList3.aspx sayfasının tarayıcıdaki görüntüsü

<script> etiketleri arasında önce tüm sayfa için geçerli olacak değişkenler (baslik, mesaj, iller) tanımlanmış ve ilk değerleri verilmiştir.

<form> etiketleri arasında, ilk satırda **baslik** değişkeninin formun içine doğrudan veri bağlama ifadesiyle bağlanacağı bildirilmiştir.

Ardından label kontrolünün text özelliğine **mesaj** değişkeni bağlanmıştır. Böylelikle, label kontrolü "mesaj" değişkeninden gelen değeri sayfada görüntülemiştir.

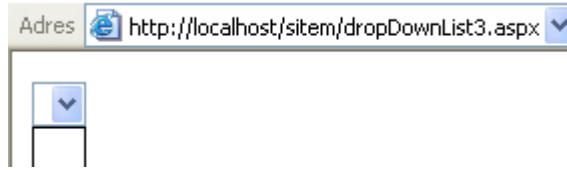
Son olarak **liste** isimli DropDownList'e **iller** dizisi bağlanmıştır.

Page_Load alt programı içinde **page.DataBind** metodu çağrılarak tüm veri kaynaklarının ilişkilendirildikleri sunucu kontrollerine bağlanmaları sağlanmıştır. Page.DataBind genellikle Page_Load olayından çağrılır. Eğer tüm sayfadaki kontrollere değil de belirli bir kontrole veri bağlanmak istenirse **kontrol.DataBind** şeklindeki kullanım tercih edilmelidir. Örneğin, sadece DropDownList kontrolüne veri bağlamak için **liste.DataBind** satırı kullanılır.

Veri kontrole bağlandıktan sonra verinin kullanıcıya nasıl görüneceği kontrolün özellikleri ayarlanarak belirlenir.

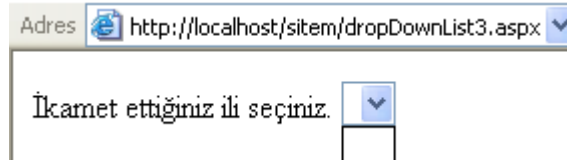
Uygulama

- a) **Page.DataBind** satırını silip sayfayı tekrar çağırıp sonucu yorumlayınız.



Resim 2.32: DropDownList3.aspx sayfasının tarayıcıdaki görüntüsü

- b) **Page.DataBind** satırını **label1.DataBind** şeklinde değiştirdikten sonra sayfayı çağırıp sonucu yorumlayınız.



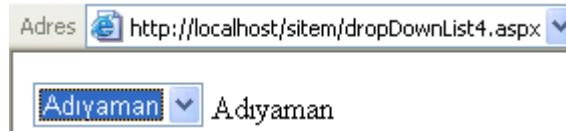
Resim 2.33: DropDownList3.aspx sayfasının tarayıcıdaki görüntüsü

dropDownList4.aspx

```

<script runat="server">
dim iller() as string = {"Adana","Adiyaman","Afyon","Ađrı","Amasya"}
Sub Page_Load(Src As Object, E As EventArgs)
If Not IsPostBack Then
    liste.DataSource = iller
End If
    page.DataBind
End Sub
</script>
</head>
<body>
<form runat="server" name="form1" method="post" action="">
<asp:DropDownList AutoPostBack="true" ID="liste" runat="server">
</asp:DropDownList>
<asp:Label ID="labell" runat="server" Text="<%=# liste.SelectedItem.Text %>">
</asp:Label>
</form>

```



Resim 2.34: DropDownList3.aspx sayfasının tarayıcıdaki görüntüsü

<script> etiketleri arasında önce tüm sayfa için geçerli olacak **iller** isimli dizi değişkeni tanımlanmış ve ilk değerleri verilmiştir.

Not IsPostBack ifadesi kullanılarak sayfanın ilk defa yüklendiğinde yapılacak işlemler belirtilmiştir. **Liste.DataSource = iller** satırıyla **liste** isimli DropDownList'in veri kaynağının **iller** dizisi olduğu belirtilmiştir.

<form> etiketleri arasında label kontrolünün text özelliğine liste isimli kontrolde tıklanan seçeneğin text özelliğinin bağlanacağı bildirilmiştir.

2.7.8. ListBox

Çeşitli seçenekleri listeler. **SelectionMode** özelliği kullanılarak birden fazla seçeneği seçmek mümkündür.

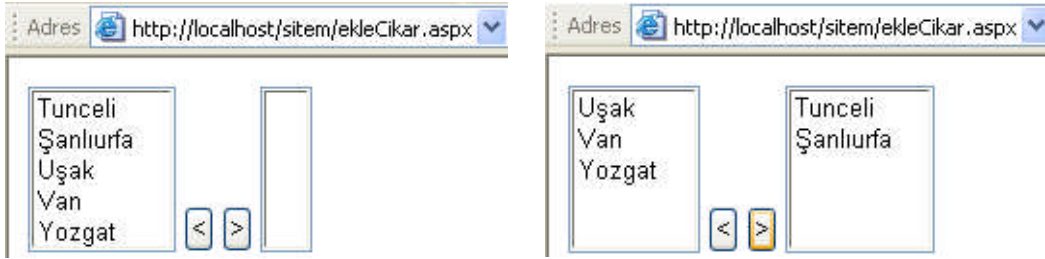
ekleCikar.aspx

```

<script language="vb" runat="server">
sub page_load(source as object, e as eventargs)
if not page.ispostback then
    with lstsolliste.items
        .add("Tunceli")
        .add("Şanlıurfa")
        .add("Uşak")
        .add("Van")
        .add("Yozgat")
    end with
end if
end sub
sub ekle(sender as object, e as eventargs)
    if lstsolliste.selectedindex = -1 then
        exit sub
    end if
    dim solmetin as string
    solmetin = lstsolliste.selecteditem.text
    if not solmetin = "" then
        lstsolliste.items.remove(solmetin)
        lstsagliste.items.add(solmetin)
    end if
end sub
sub cikar(sender as object, e as eventargs)
    if lstsagliste.selectedindex = -1 then
        exit sub
    end if
    dim sagmetin as string
    sagmetin = lstsagliste.selecteditem.text
    if not sagmetin = "" then
        lstsagliste.items.remove(sagmetin)
        lstsolliste.items.add(sagmetin)
    end if
end sub
</script>

<form runat="server" name="form1" method="post" action="">
<asp:ListBox ID="lstsolliste" Rows="5" runat="server"></asp:ListBox>
<asp:Button ID="btncikart" runat="server" Text="<" OnClick="cikar" />
<asp:Button ID="btnekle" runat="server" Text=">" OnClick="ekle" />
<asp:ListBox ID="lstsagliste" Rows="5" runat="server"></asp:ListBox>
</form>

```

Resim 2.35: EkleCikar.aspx sayfasının tarayıcıdaki görüntüsü

Sayfa ilk yüklendiğinde **with...end with** bloğu kullanılarak **lstsolliste** isimli **listbox**'a il isimleri eklenmiştir.

Ekle alt programı **lstsolliste**'den **lstsagliste**'ye illeri aktarmak için, **cikar** alt programı **lstsagliste**'deki illeri **lstsolliste**'ye geri göndermek, yani **lstsagliste**'den çıkarmak için oluşturulmuştur.

Eğer **lstsolliste**'de bir seçenek seçilmemişse (**lstsolliste.selectedindex = -1**) alt programdan çıkılmaktadır (**exit sub**). Fakat bir seçenek seçildiğinde **lstsolliste**'de seçilen il (**lstsolliste.selecteditem.text**) **solmetin** adlı **string** değişkene aktarılmaktadır.

Solmetin değişkeni boş olmadığında (**not solmetin = ""**) **solmetin** değeri soldaki listeden çıkarılıp sağdaki listeye eklenmektedir.

Uygulama

Butonlara tıklamadan **lstsolliste**'de bir il seçildiğinde o ili **otomatik** olarak **lstsolliste**'den çıkarıp **lstsagliste**'ye ekleyecek ve tam tersi işlemi yapacak şekilde **eklecikar.aspx** sayfasını değiştiriniz.

2.7.9. RadioButton

Tek bir radyo düğmesi ekler. **Groupname** özelliği ile radyo düğmeleri gruplandırılabilir. Böylelikle bir grup seçenek içinden yalnızca birinin işaretlenmesi sağlanmış olur.

degerlendir.aspx

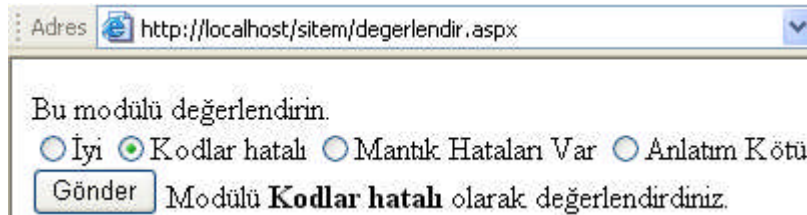
```
<script runat="server">
sub gonder(sender as object, e as eventargs)
    if radiol.checked="True" then
        mesaj.text = "Modülü " & "<b>" & radiol.text & "</b>" _
```

```

    & " olarak değerlendirdiniz."
else if radio2.checked="True" then
    mesaj.text = "Modülü " & "<b>" & radio3.text & "</b>" _
    & " olarak değerlendirdiniz."
else if radio3.checked="True" then
    mesaj.text = "Modülü " & "<b>" & radio4.text & "</b>" _
    & " olarak değerlendirdiniz."
else
    mesaj.text = "Modülü " & "<b>" & radio5.text & "</b>" _
    & " olarak değerlendirdiniz."
end if
end sub
</script>

<form runat="server" name="form1" method="post" action="">
Bu modülü değerlendirin.<br>
<asp:RadioButton ID="radiol" runat="server" Text="İyi"
groupname="degerlendirme" Checked="true" />
<asp:RadioButton ID="radio2" runat="server" Text="Kodlar hatalı"
groupname="degerlendirme" />
<asp:RadioButton ID="radio3" runat="server" Text="Mantık Hataları Var"
groupname="degerlendirme" />
<asp:RadioButton ID="radio4" runat="server" Text="Anlatım Kötü"
groupname="degerlendirme" /><br>
<asp:Button ID="btngonder" runat="server" Text="Gönder" OnClick="gonder" />
<asp:Label ID="mesaj" runat="server"></asp:Label>
</form>

```



Resim 2.36: Degerlendir.aspx sayfasının tarayıcıdaki görüntüsü

2.7.10. RadioButtonList

RadioButtonList, aynı anda sadece biri seçilebilen bir grup seçenek oluşturur.

alisveris.aspx

```

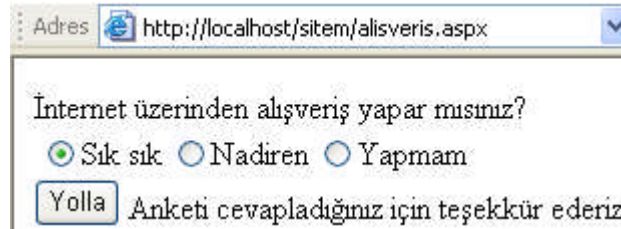
<script runat="server">
sub yolla(sender as object, e as EventArgs)
    mesaj.text="Anketi cevapladığınız için teşekkür ederiz."
end sub
</script>

```

```

<form runat="server" name="form1" method="post" action="">
İnternet üzerinden alışveriş yapar mısınız?
<asp:RadioButtonList ID="alisveris" runat="server" RepeatDirection="Horizontal">
<asp:ListItem>Sık sık</asp:ListItem>
<asp:ListItem>Nadiren</asp:ListItem>
<asp:ListItem>Yapmam</asp:ListItem>
</asp:RadioButtonList>
<asp:Button ID="btnyolla" runat="server" Text="Yolla" OnClick="yolla" />
<asp:Label ID="mesaj" runat="server"></asp:Label>
</form>

```



Resim 2.37: Alisveris.aspx sayfasının tarayıcıdaki görüntüsü

2.7.11. Table, TableRow, TableCell

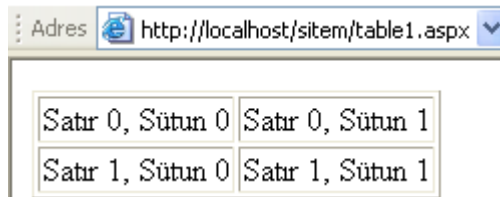
Tablo oluşturulmasını sağlayan kontroldür. Kontrol **Tag Chooser** penceresinden eklenebilir.

table1.aspx

```

<form runat="server" name="form1" method="post" action="">
<asp:Table ID="Tablo" runat="server" gridlines="Both">
  <asp:TableRow>
    <asp:TableCell>Satır 0, Sütun 0</asp:TableCell>
    <asp:TableCell>Satır 0, Sütun 1</asp:TableCell>
  </asp:TableRow>
  <asp:TableRow>
    <asp:TableCell>Satır 1, Sütun 0</asp:TableCell>
    <asp:TableCell>Satır 1, Sütun 1</asp:TableCell>
  </asp:TableRow>
</asp:Table>
</form>

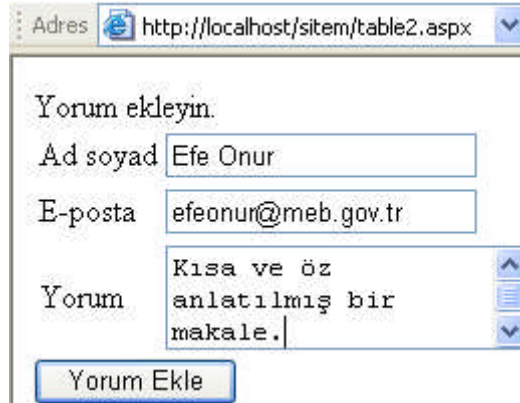
```



Resim 2.38: Table1.aspx sayfasının tarayıcıdaki görüntüsü

table2.aspx (Kontrol içinde kontrol kullanma örneği)

```
<form runat="server" name="form1" method="post" action="">
Yorum ekleyin.
<asp:Table ID="Tablo" runat="server">
  <asp:TableRow>
    <asp:TableCell>Ad soyad</asp:TableCell>
    <asp:TableCell>
      <asp:textbox id="txtad" runat="server"/>
    </asp:TableCell>
  </asp:TableRow>
  <asp:TableRow>
    <asp:TableCell>E-posta</asp:TableCell>
    <asp:TableCell>
      <asp:textbox id="txtposta" runat="server"/>
    </asp:TableCell>
  </asp:TableRow>
  <asp:TableRow>
    <asp:TableCell>Yorum</asp:TableCell>
    <asp:TableCell>
      <asp:textbox id="txtyorum" runat="server"
        textmode="multiline" rows=3/>
    </asp:TableCell>
  </asp:TableRow>
</asp:Table>
<asp:Button ID="btnyorum" runat="server" Text="Yorum Ekle"
  OnClick="ekle" />
<asp:Label ID="mesaj" runat="server"></asp:Label>
</form>
```



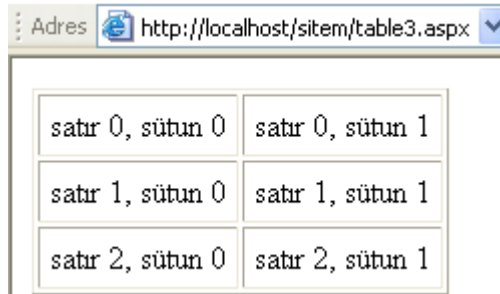
Resim 2.39: Table2.aspx sayfasının tarayıcıdaki görüntüsü

Table3.aspx sayfasında programatik olarak (dinamik olarak) çalışma zamanında bir table kontrolü oluşturulmaktadır.

table3.aspx

```
<script runat="server">
Sub Page_Load(Src As Object, E As EventArgs)
dim satirlar,hucrerler,i,j as byte
satirlar=3
hucrerler=2
For i=0 To satirlar-1
    dim tsatir As New TableRow()
    For j=0 To hucrerler-1
        dim tsutun As New TableCell()
        tsutun.Controls.Add(New LiteralControl("sadır " & i & ", sütün " & j))
        tsatir.Cells.Add(tsutun)
    Next
    Table1.Rows.Add(tsatir)
Next
End Sub
</script>

<form runat="server" name="form1" method="post" action="">
<asp:Table ID="table1" runat="server" CellPadding="5" GridLines="Both">
</asp:Table>
</form>
```



Resim 2.40: Table3.aspx sayfasının tarayıcıdaki görüntüsü

2.7.12. Literal

Sayfa üzerinde metin görüntülemek için kullanılır. İçeriğine stil uygulanamaz.

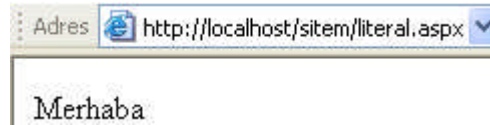
literal.aspx

```
<script runat="server">
Sub Page_Load(Src As Object, E As EventArgs)
literall.text = "Merhaba"
End Sub
</script>
```

```

<form runat="server" name="form1" method="post" action="">
<asp:Literal ID="literall" runat="server" />
</form>

```



Resim 2.41: Literal.aspx sayfasının tarayıcıdaki görüntüsü

2.7.13. Panel

Sayfa üzerindeki kontrolleri gruplandırmak için kullanılır. Diğer kontroller için bir barındırıcı (**container**) görevi görür. Panel kontrolü sayesinde bir grup kontrolün gizlenmesi veya gizli hâldeyken gösterilmesi sağlanır. Bu işlem Panel kontrolünün **visible** özelliği kullanılarak gerçekleştirilir.

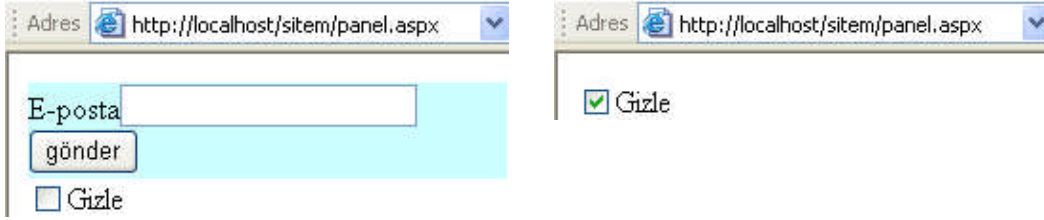
Panel.aspx

```

<script runat="server">
Sub Page_Load(Src As Object, E As EventArgs)
panell.Controls.Add(New LiteralControl("E-posta"))
dim textbox1 as textbox = new textbox
panell.Controls.Add(textbox1)
panell.Controls.Add(new LiteralControl("<br>"))
dim buton1 as button = new button
panell.Controls.Add(buton1)
buton1.text="gönder"
buton1.id = "gonder"
End Sub
Sub gizle(sender as object, e as eventargs)
if checkbox1.checked then
panell.visible = false
else
panell.visible = true
end if
End Sub
</script>

<form runat="server" name="form1" method="post" action="">
<asp:Panel ID="panell" runat="server" BackColor="#CCFFFF"
Height="50px" Width="250" ></asp:Panel>
<asp:CheckBox AutoPostBack="true" ID="checkboxox1" runat="server"
Text="Gizle" OnCheckedChanged="gizle" />
</form>

```



Resim 2.42: Panel.aspx sayfasının tarayıcıdaki görüntüsü

2.7.14. AdRotator

Bu kontrol reklam resimleri görüntüler. Resimlere bağlantı (link) verilebilir. Sayfanın her açılışında veya sayfa güncellendiğinde önceden tanımlanmış resim listesinden rastgele bir resim görüntülenebilir.

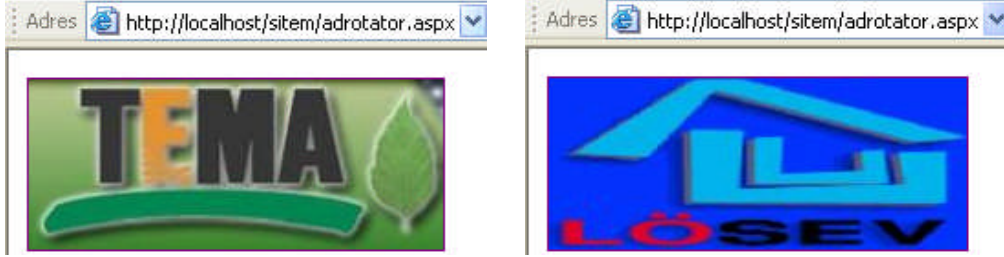
Bu kontrolün reklam bilgileri XML dosyasında tutulur. AdRotator bu dosyaya bağlanarak belirtilen reklamları görüntüler. Biz ticari bir reklam yerine toplum yararına hizmet veren iki vakfı tanıtan bir örnek yapalım. Önce XML dosyasını, ardından sayfayı oluşturalım.

File menüsünden **New** komutunu kullanarak **New Document** penceresine geliniz. **General** sekmesinde **Basic Page** kategorisinden **XML** seçeneğini seçip **Create** düğmesine tıklayınız. Aşağıdaki satırları yazarak dosyayı **tanitim.xml** adıyla kaydediniz.

```
<?xml version="1.0" encoding="utf-8"?>
<Advertisements>
  <Ad>
    <ImageUrl>resimler/tema.jpg</ImageUrl>
    <NavigateUrl>http://www.tema.org.tr</NavigateUrl>
    <AlternateText>Türkiye Erozyonla Mücadele, Ağaçlandırma ve
    Doğal Varlıkları Koruma Vakfı</AlternateText>
    <Impressions>50</Impressions>
  </Ad>
  <Ad>
    <ImageUrl>resimler/losev.jpg</ImageUrl>
    <NavigateUrl>http://www.losev.org.tr</NavigateUrl>
    <AlternateText>Lösemili Çocuklar Vakfı</AlternateText>
    <Impressions>50</Impressions>
  </Ad>
</Advertisements>
```

adRotator.aspx

```
<form runat="server" name="form1" method="post" action="">
<asp:AdRotator AdvertisementFile="tanitim.xml" BorderWidth="1"
ID="reklam" runat="server" Target="_blank" />
</form>
```



Resim 2.43: Adrotator.aspx sayfasının tarayıcıdaki görüntüsü

XML dosyasındaki etiketlerin ne anlama geldiklerini inceleyelim.

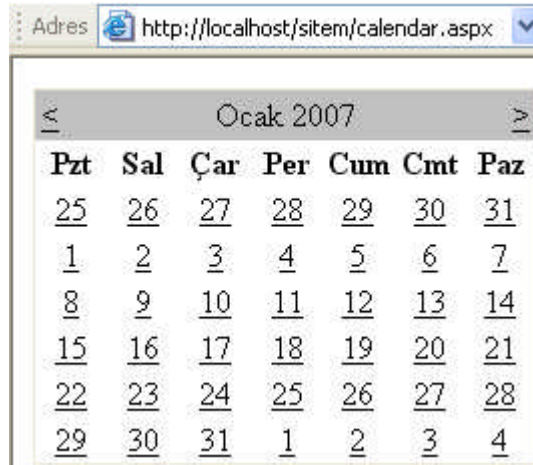
Dosyadaki tüm reklam bilgileri `<Advertisements></Advertisements>` etiketleri arasında, her bir reklam bilgisi ise `<Ad></Ad>` etiketleri arasında yer almaktadır. `<ImageUrl></ImageUrl>` etiketleri arasında resim dosyasının adres tanımı, `<NavigateUrl></NavigateUrl>` etiketleri arasında ise resme tıklandığında gösterilecek sayfanın URL'si belirtilmektedir. Resmin yüklenememesi durumunda veya üzerine gelindiğinde gösterilecek yazı `<AlternateText></AlternateText>` etiketleri arasında, resmin diğer resimler arasında gösterilme sıklığı `<Impressions></Impressions>` etiketleri arasında tanımlanır. Resmin ne kadar sık görüntülenmesi isteniyorsa `<Impressions>` etiketleri arasındaki sayı o kadar büyük verilmelidir.

2.7.15. Calendar

Takvim ekler. Özellikleri kullanılarak farklı görünümde takvimler oluşturulabilir.

Calendar.aspx

```
<form runat="server" name="form1" method="post" action="">
<asp:Calendar ID="calendar1" runat="server"></asp:Calendar>
</form>
```



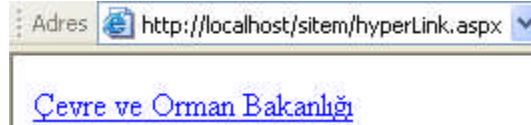
Resim 2.44: Calendar.aspx sayfasının tarayıcıdaki görüntüsü

2.7.16. HyperLink

Metin veya resim linki oluşturur. Resim linki oluşturmak için **ImageUrl** özelliği ayarlanır. Text ve ImageUrl özelliklerinin her ikisi de ayarlanırsa resim linki önceliği alır, metin ise araç ipucu (**Tool Tip**) görevini üstlenir. Resim görüntülenemezse metin görüntülenir.

hyperLink.aspx

```
<form runat="server" name="form1" method="post" action="">
<asp:HyperLink Runat="Server" Text="Çevre ve Orman Bakanlığı"
  NavigateUrl="http://www.cevreorman.gov.tr/"
  Target="_blank"/>
</form>
```



Resim 2.45: HyperLink.aspx sayfasının tarayıcıdaki görüntüsü

2.7.17. Image

Resim görüntüler.

```
<script runat="server">
Sub goster(Src As Object, E As EventArgs)
  image1.ImageUrl = "resimler/" & dropdown1.SelectedItem.Value
  image1.AlternateText = dropdown1.SelectedItem.Text
End Sub
</script>

<form runat="server" name="form1" method="post" action="">
<asp:Image ID="image1" ImageUrl="resimler/federasyon.gif"
  AlternateText="Türkiye Futbol Federasyonu" runat="server" /><br />
  Amblemini görmek istediğiniz takımı seçiniz.
<asp:DropDownList AutoPostBack="true" ID="dropdown1" runat="server"
  OnSelectedIndexChanged="goster">
  <asp:ListItem Value="bjk.gif">Beşiktaş</asp:ListItem>
  <asp:ListItem Value="gs.gif">Galatasaray</asp:ListItem>
  <asp:ListItem Value="fb.gif">Fenerbahçe</asp:ListItem>
  <asp:ListItem Value="sivasspor.gif">Sivasspor</asp:ListItem>
</asp:DropDownList>
</form>
```



Resim 2.46: Image.aspx sayfasının tarayıcıdaki görüntüsü

2.8. Geçerlilik Sunucu Kontrolleri

Kullanıcı bilgilerinin istenilen kriterlere uyup uymadığını denetlemek amacıyla kullanılan kontrollerdir. ÖrneĐin, kullanıcının bir alanı boş bırakmasını engellemek veya şifre için minimum karakter sayısı belirlemek gibi. Geçerlilik kontrolleri ve görevleri aşağıda belirtilmiştir.

RequiredFieldValidator: Bir kontrole bilgi girilmesini zorunlu kılar.

CompareValidator: Kullanıcının girdiĐi deĐer ile başka bir deĐeri karşılaştırmak için kullanılır.

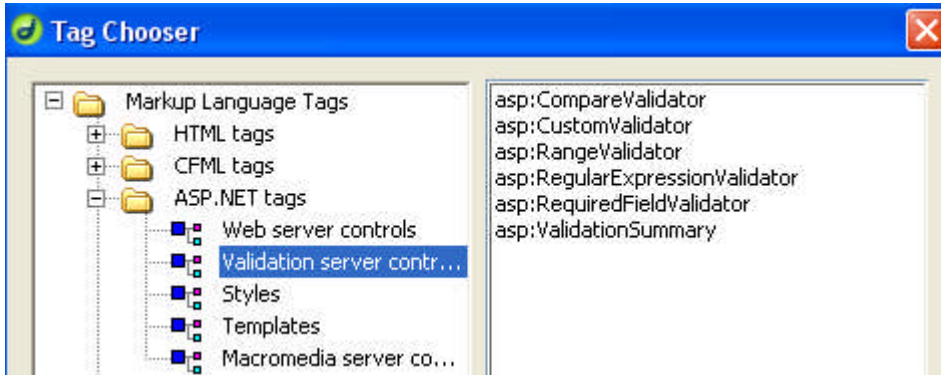
RangeValidator: Kullanıcının belirtilen aralıkta deĐerler girmesini sağlar.

RegularExpressionValidator: Bilginin belli bir biçimde girilmesi sağlar.

CustomValidator: Web geliştiricinin istediĐi bir geçerlilik denetleme yöntemi belirlemesini sağlar. ÖrneĐin, sadece tek sayıların veya çift sayıların girişine izin verme gibi.

ValidationSummary: GeçerliliĐi kontrollerinin gerçekleştirdiĐi geçerlilik işlemleri sonucunda oluşan hatalar ve hata mesajları gibi bilgileri tutar.

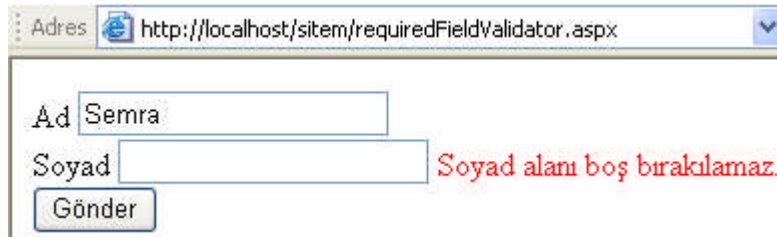
Tag Chooser penceresindeki **Validation Server Controls** bölümünden istenilen geçerlilik sunucu kontrolü eklenebilir.



Resim 2.47: Validation Server Controls

requiredFieldValidator.aspx

```
<form runat="server" name="form1" method="post" action="">
<asp:Label ID="label1" runat="server" Text="Ad"></asp:Label>
<asp:TextBox ID="textbox1" runat="server" />
<asp:RequiredFieldValidator ControlToValidate="textbox1"
ErrorMessage="Ad alanı boş bırakılamaz." ID="rfvalidator1" runat="server"/><br />
<asp:Label ID="label2" runat="server" Text="Soyad"></asp:Label>
<asp:TextBox ID="textbox2" runat="server" />
<asp:RequiredFieldValidator ControlToValidate="textbox2"
ErrorMessage="Soyad alanı boş bırakılamaz." ID="rfvalidator2" runat="server"/><br />
<asp:Button ID="button1" runat="server" Text="Gönder" />
</form>
```

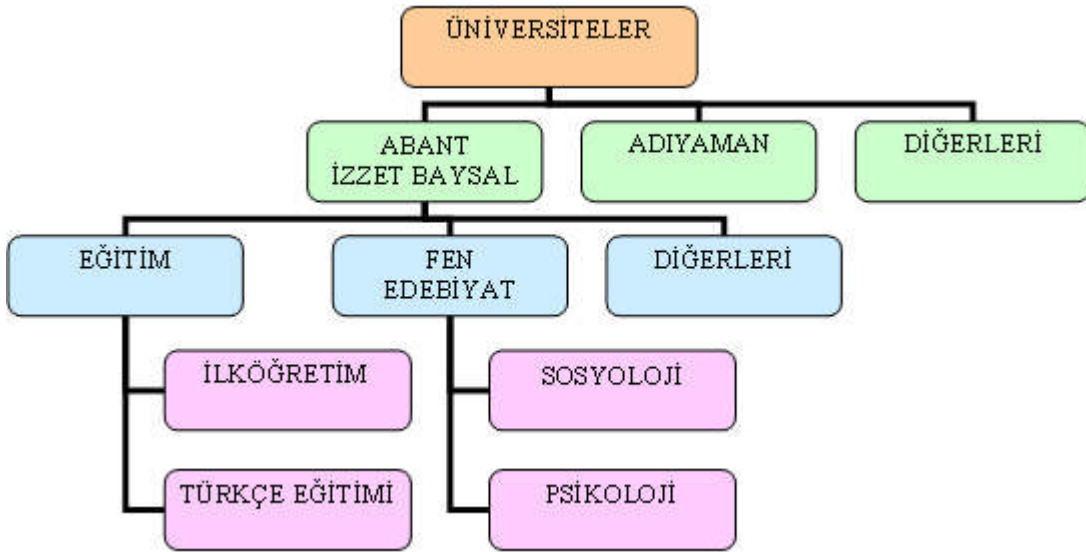


Resim 2.48: requiredFieldValidator.aspx ekran görüntüsü

UYGULAMA FAALİYETİ

Mezun olunan bölümün sırasıyla üniversitelerin, fakültelerin ve bölümlerin listelendiği ListBox'lerden seçildiği ve seçimin tarayıcıya yazdırıldığı sayfayı geliştiriniz.

NOT: Bu tür uygulamalarda veriler veritabanlarında veya XML belgelerinde tutulur ve verilere ihtiyaç duyulduğunda bu veri kaynaklarından veriler çekilir. Veritabanı ve XML kavramlarını henüz öğrenmediğiniz için verileri şimdilik dizilerde tutunuz.



Adres <http://localhost/sitem/universiteler.aspx>

Mezun olduğunuz bölümü seçiniz :

ABANT İZZET BAYSAL	EĞİTİM	İLKÖĞRETİM
ADYAMAN	FEN EDEBİYAT	TÜRKÇE EĞİTİMİ

ADYAMAN ÜNİVERSİTESİ
EĞİTİM FAKÜLTESİ
TÜRKÇE EĞİTİMİ BÖLÜMÜNDEN MEZUNSUNUZ.

İşlem Basamakları	Öneriler
Verileri tutmak için dizileri tanımlayınız ve değerlerini atayınız.	Dim universiteler() As String = {"ABANT İZZET BAYSAL", "ADYAMAN"}
Değerleri ListBox'a aktarmak için kullanılacak değişkeni tanımlayınız.	Dim i As String

Sayfa ilk yüklendiğinde üniversite isimlerini ListBox1'e aktarınız. ListBox1'de üniversite isimleri tutulmaktadır.	<pre> Sub Page_Load(...) If Not IsPostBack Then For Each i in universiteler ListBox1.Items.Add(i) Next End If End Sub </pre>
Hangi üniversite seçildiyse o üniversitenin fakültelerini ListBox2'ye ekleyiniz. ListBox2'ye önceden eklenmiş olabilecek fakülteleri silmeyi unutmayınız. ListBox2'de fakülte isimleri tutulmaktadır.	<pre> Sub ListBox1_SelectedIndexChanged(...) If ListBox1.SelectedValue = "ABANT İZZET BAYSAL" ListBox2.Items.Clear() </pre>
Hangi fakülte seçildiyse o fakültenin bölümlerini ListBox3'ye ekleyiniz. ListBox3'e önceden eklenmiş olabilecek bölümleri silmeyi unutmayınız. ListBox3'de bölüm isimleri tutulmaktadır.	<pre> Sub ListBox2_SelectedIndexChanged(...) If (ListBox1.SelectedValue = "ABANT İZZET BAYSAL") AND (ListBox2.SelectedValue = "EĞİTİM") ListBox3.Items.Clear() </pre>
Seçilen bölümü görüntüleyiniz.	Label1.Text.....

Universiteler.aspx

```

7 <script runat="server">
8 Dim universiteler() As String = {"ABANT İZZET BAYSAL","ADIYAMAN"}
9 Dim abant_fakulte() As String = {"EĞİTİM","FEN EDEBİYAT"}
10 Dim adiyaman_fakulte() As String = {"EĞİTİM","FEN EDEBİYAT"}
11 Dim abant_egitim_bolum() As String = {"İLKÖĞRETİM","TÜRKÇE EĞİTİMİ"}
12 Dim adiyaman_egitim_bolum() As String = {"İLKÖĞRETİM","TÜRKÇE EĞİTİMİ"}
13 Dim abant_fenedebiyat_bolum() As String = {"PSİKOLOJİ","SOSYOLOJİ"}
14 Dim adiyaman_fenedebiyat_bolum() As String = {"PSİKOLOJİ","SOSYOLOJİ"}
15 Dim i As String
16
17 Sub Page_Load(Src As Object, E As EventArgs)
18 If Not IsPostBack Then
19     For Each i in universiteler
20         ListBox1.Items.Add(i)
21     Next
22 End If
23 End Sub

```

```

24
25 Sub ListBox1_SelectedIndexChanged(Src As Object, E As EventArgs)
26 If ListBox1.SelectedValue = "ABANT İZZET BAYSAL"
27     ListBox2.Items.Clear()
28     For Each i in abant_fakulte
29         ListBox2.Items.Add(i)
30     Next
31 End If
32 If ListBox1.SelectedValue = "ADIYAMAN"
33     ListBox2.Items.Clear()
34     For Each i in adiyaman_fakulte
35         ListBox2.Items.Add(i)
36     Next
37 End If
38 End Sub
39
40 Sub ListBox2_SelectedIndexChanged(Src As Object, E As EventArgs)
41 If (ListBox1.SelectedValue = "ABANT İZZET BAYSAL") AND
42     (ListBox2.SelectedValue = "EĞİTİM")
43     ListBox3.Items.Clear()
44     For Each i in abant_egitim_bolum
45         ListBox3.Items.Add(i)
46     Next
47 End If
48 If (ListBox1.SelectedValue = "ABANT İZZET BAYSAL") AND
49     (ListBox2.SelectedValue = "FEN EDEBİYAT")
50     ListBox3.Items.Clear()
51     For Each i in abant_fenedebiyat_bolum
52         ListBox3.Items.Add(i)
53     Next
54 End If
55 If (ListBox1.SelectedValue = "ADIYAMAN") AND
56     (ListBox2.SelectedValue = "EĞİTİM")
57     ListBox3.Items.Clear()
58     For Each i in adiyaman_egitim_bolum
59         ListBox3.Items.Add(i)
60     Next
61 End If
62 If (ListBox1.SelectedValue = "ADIYAMAN") AND
63     (ListBox2.SelectedValue = "FEN EDEBİYAT")
64     ListBox3.Items.Clear()
65     For Each i in adiyaman_fenedebiyat_bolum
66         ListBox3.Items.Add(i)
67     Next

```

```

68 End If
69 End Sub
70
71 Sub ListBox3_SelectedIndexChanged(Src As Object, E As EventArgs)
72     Label1.Text = ListBox1.SelectedValue & " ÜNİVERSİTESİ" & "<br/>" & _
73     ListBox2.SelectedValue & " FAKÜLTESİ " & "<br/>" & _
74     ListBox3.SelectedValue & " BÖLÜMÜNDEN MEZUNSUNUZ."
75 End Sub
76 </script>
77 </head>
78 <body>
79 <form runat="server" name="form1" method="post" action="">
80 Mezun olduğunuz bölümü seçiniz : <br/>
81 <asp:ListBox ID="ListBox1" runat="server" AutoPostBack="true"
82 OnSelectedIndexChanged="ListBox1_SelectedIndexChanged"/>
83 <asp:ListBox ID="ListBox2" runat="server" AutoPostBack="true"
84 OnSelectedIndexChanged="ListBox2_SelectedIndexChanged"/>
85 <asp:ListBox ID="ListBox3" runat="server" AutoPostBack="true"
86 OnSelectedIndexChanged="ListBox3_SelectedIndexChanged"/><br/>
87 <asp:Label ID="label1" runat="server"></asp:Label>
88 </form>

```

ÖLÇME VE DEĞERLENDİRME

Aşağıda verilen sorular için doğru cevap seçeneklerini işaretleyiniz.

1. CheckBox kontrolü işaretlendiğinde veya işareti kaldırıldığında oluşan olay nedir?
A) OnClick
B) OnLoad
C) OnTextChanged
D) OnCheckedChanged
2. DropDownList kontrolünde seçilen seçeneğin değişmesiyle oluşan olay nedir?
A) OnInit
B) OnDispose
C) OnSelectedIndexChanged
D) OnCommand
3. Kontrolün veri kaynağını belirlemek için hangi özellik kullanılır?
A) Data
B) DataSource
C) DataMember
D) DataTextField
4. Sayfa üzerindeki kontrolleri gruplandırmak için kullanılan kontrol nedir?
A) Calendar
B) Panel
C) Image
D) DropDownList
5. Kontrollerin görünüp görünmemesini sağlayan özellik nedir?
A) Visible
B) Property
C) Index
D) Checked

ÖĞRENME FAALİYETİ-3

AMAÇ

Veritabanı bağlantısını gerçekleştirerek verileri sitenizde görüntüleyebilecek ve veriler üzerinde yaptığımız değişikliklerle veritabanınızı güncelleyebileceksiniz.

ARAŞTIRMA

Veritabanından verilerin alınıp görüntülediği ve veritabanındaki verilerin kullanıcının girdiği verilerle güncellendiği örnek siteler bulunuz.

3. ADO.NET

ADO.NET (**ActiveX Data Object**), veri kaynaklarıyla bağlantı kurarak verileri almak, silmek, güncellemek, yeni veriler eklemek için kullanılan .NET sınıflarının tümüdür. Bu öğrenme faaliyetinde basit uygulamalar geliştirebilmeniz için gereken düzeyde ADO.NET kavramları anlatılacaktır. ADO.NET'e geçmeden önce veriyle ilişkili kavramları kısaca açıklayalım.

3.1. Veri (Data)

Sayısal veya mantıksal her türlü değer bir veridir. Verinin işlenmiş ve bir anlam ifade eden hâline ise bilgi adı verilir. Örneğin, saatte yelkovanın 12 üzerinde olması bir veridir. Aynı şekilde akrebin 4 üzerinde olması bir veridir. Bu iki verinin yorumlanmasıyla elde edilen "saat dörttür" ifadesi bir bilgidir.

Bir bilgi farklı durumlarda veri olarak kullanılabilir. Örneğin, bir arkadaşımızla buluşma saatinizin üç veya dört olması birer veriyken "Buluşmaya 1 saat geç kaldım" ifadesi bir bilgidir. Görüldüğü gibi "saatin dört olması" bir önceki örnekte bilgiyken bu örnekte bir veridir.

Bir kişi veya uygulama için gerekli olan bir veri veya bilgi diğer kişi veya uygulamalar için gerekli olmayabilir. Örneğin, bir alışveriş sitesi birçok kişisel bilgiye ihtiyaç duyarken bir sohbet sitesi sadece rumuz isteyip diğer bilgileri istemeyebilir.

Veri yapıları farklı şekillerde oluşturulabilir. Bir uygulamada ad soyad bilgileri birlikte tutulurken diğer bir uygulamada iki bilgi de ayrı ayrı tutulabilir. Bir başka örnek ise adres bilgilerinin tutulmasıdır. Bir uygulamada adres tek bir alanda tutulurken diğer bir uygulamada cadde, sokak, apartman nu, daire nu, ilçe, il bilgileri ayrı alanlarda tutulabilir.

3.2. Veri Kaynağı (Data Source)

Uygulamalar kullanacakları verileri çeşitli veri kaynaklarından alır. Bu veri kaynağı bir metin (**text**) dosyası, Access gibi bir veritabanı programı veya XML belgesi olabilir. Her

veri kaynağı, verileri kendine özel bir düzen içinde saklar. Bir metin dosyasında veriler satır ve sütunlardan oluşan bir düzen içinde tutulur.



Ad	soyad	Meslek
Seda	Çiçek	Avukat
Eda	Güneş	Muhasebeci
Simge	Mavi	Mühendis
Nilsu	Şirin	Doktor
Yadigar	Çiçek	Jeolog
Cansu	Deniz	Öğretmen
Gözde	Umut	Mühendis
Özge	Derya	Hemşire

Resim 3.1: Metin dosyasında verilerin bir düzen içinde tutulması

Yukarıdaki veriler bir XML belgesinde aşağıdaki gösterildiği gibi daha farklı bir düzen içinde tutulmaktadır.

```
<musteri_bilgileri>
  <kisiler>
    <ad>Seda</ad>
    <soyad>Çiçek</soyad>
    <meslek>Avukat</meslek>
  </kisiler>
  <kisiler>
    <ad>Eda</ad>
    <soyad>Güneş</soyad>
    <meslek>Muhasebeci</meslek>
  </kisiler>
  <kisiler>
    <ad>Simge</ad>
    <soyad>Mavi</soyad>
    <meslek>Mühendis</meslek>
  </kisiler>
  .....
</musteri_bilgileri>
```

Veritabanı programlarında ise veriler satır ve sütunlardan oluşan bir düzen içinde tutarlar. Ancak, veritabanlarının kendilerine has biçimsel bir yapıları vardır. Veritabanı programları, büyük miktardaki veriler üzerinde her türlü ilişkilendirmeyi, sorgulamayı, düzenlemeyi, yönetmeyi kolaylaştıran özelliklerinden dolayı tercih edilmektedir.

Ad	Soyad	Meslek
Seda	Çiçek	Avukat
Eda	Güneş	Muhasebeci
Simge	Mavi	Mühendis
Nilsu	Şirin	Doktor
Yedigâr	Çiçek	Jeolog
Cansu	Deniz	Öğretmen

Resim 3.2: Veritabanında verilerin bir düzen içinde tutulması

3.3. Veritabanı Kavramları

Modül boyunca kullanılacak Access programından hareketle veritabanı kavramlarını kısaca değinelim. Veritabanları programlarında veriler tablolar içinde tutulur. Tablo (**table**), satır (**row**) ve sütunlardan (**column**) oluşur. Her sütunda sütun başlığında belirtilen veriler tutulur. Örneğin, başlığında “Meslek” yazan sütunda meslek bilgileri tutulur.

Satır ve sütunların kesişmesinden oluşan hücelere “alan” (**field**) adı verilir. Veriler bu alanlara girilir. Girilecek veriye göre alanın biçimi ve türü belirlenir. Örneğin, doğum tarihlerini girmek için “Tarih/Saat” türü ve “Kısa Tarih – 6/19/1994” biçimi seçilir.

Ad	Soyad	Meslek
Seda	Çiçek	Avukat
Eda	Güneş	Muhasebeci
Simge	Mavi	Mühendis
Nilsu	Şirin	Doktor
Yedigâr	Çiçek	Jeolog
Cansu	Deniz	Öğretmen

Resim 3.3: Veritabanı programında alan, satır, kayıt, sütun kavramları

Veritabanlarında arama ve sıralama amacıyla bir alanın aldığı değerlerin birbirinden farklı olmasını sağlamak için birincil anahtar (**primary key**) atanır. Örneğin, kimlik numaralarının birbirinden farklı olması için bu alan birincil anahtar olarak ayarlanır.

Veritabanlarında birden çok tablo bulunabilir. Bir tablodaki alanla ilişkili veriler diğer bir tabloda verilmiş olabilir. Bu durumda tablolar arasında ilişki diğer bir ifadeyle bağlantı (**relationship**) kurularak bir alana bağlı diğer tablolardaki verilere ulaşılır.

Bir veritabanı programını az da olsa kullanarak veritabanı kavramlarına hâkim olmanız veri alışverişi için kullanılan ADO.NET ve web sunucu kontrollerini daha iyi anlamanızı sağlayacaktır.

3.4. SQL (Structured Query Language)

SQL, veritabanlarıyla veri alışverişi için geliştirilmiş standart bir dildir. Bu dil kullanılarak aşağıdaki veritabanı işlemleri yapılabilir:

- Veri eklemek.
- Verileri almak.
- Sorgulama yapmak.
- Verileri silmek.
- Verileri güncellemek.

Bu bölümde modülün ilerleyen sayfalarında kullanılacak SQL deyimleri kısaca anlatılacaktır. Bu deyimlerin çalışması prensibi anlatılırken **Kisiler** tablosu kullanılacaktır.

3.4.1. SELECT Deyimi

Veritabanından verileri almak için kullanılır. Bu deyimle birlikte kullanılan koşula uygun veriler veritabanından elde edilir. Kullanım şekli;

```
SELECT sütun ismi/isimleri FROM tablo ismi [WHERE arama koşulu]
[ORDER BY sıralanış bildirimi ASC veya DESC]
```

NOT: SQL deyimleri büyük küçük harfe duyarlı değildir. Deyimleri büyük harfle (SELECT) veya küçük harfle (select) yazmak arasında herhangi bir fark yoktur.

Kullanım şeklindeki köşeli parantez içindeki ifadelerin kullanılması zorunlu değildir, isteğe bağlıdır. Aşağıda bu deyim kullanımıyla ilgili örnekler verilmiştir.

SELECT * FROM Kisiler	Kisiler tablosundan tüm verileri getirir.
SELECT Ad FROM Kisiler	Ad sütunundaki alanları getirir.
SELECT Ad,Soyad FROM Kisiler	Ad ve Soyad sütunlarındaki alanları getirir.
SELECT * FROM Kisiler WHERE Ad="Cansu"	Ad sütunu alanlarında Cansu ifadesini barındıran tüm kayıtları getirir. Kısaca ifade edilirse tabloda adı Cansu olan kayıtları getirir.
SELECT * FROM Kisiler WHERE Soyad="Çiçek"	Soyadı Çiçek olan kayıtları getirir.

Tablo 3.1: Select deyiminin kullanımıyla ilgili örnekler

Uygulama

- a) Kisiler tablosundan sadece **Soyad** sütunundaki alanları getiren,
- b) Sadece **Meslek** sütunundaki alanları getiren,

c) Tablodan **Ad, Soyad, Meslek** sütunlarındaki alanları getiren SELECT deyimlerini yazınız.

3.4.2. INSERT Deyimi

Veritabanına yeni bir kayıt eklemek için kullanılır. Kullanım şekli şöyledir:

```
INSERT INTO tablo ismi [(verinin ekleneceği sütun isimleri)]  
VALUES (eklenecek değerler)
```

```
INSERT INTO Kisiler (Ad,Soyad) VALUES ("Gül","Naz")
```

ifadesi, Kisiler tablosunun Ad ve Soyad alanlarına yeni değerleri eklemeyi sağlar. Böylelikle tabloya yeni bir kayıt eklenmiş olur.

Uygulama

Tabloya 3 yeni kayıt ekleyiniz.

3.4.3. UPDATE Deyimi

Koşul veya koşullar belirtilerek kayıtların güncellenmesini sağlayan deyimdir. Kullanım şekli;

```
UPDATE tablo ismi SET sütun ismi [WHERE arama koşulu]
```

```
UPDATE Kisiler SET Ad="Cem" WHERE Soyad="Çiçek"
```

ifadesi soyadı "Çiçek" olan kayıtların Ad verisini "Cem" olarak değiştirmeyi sağlar. Sonuç olarak tablodaki "Yadigar" verisini "Cem" olarak değiştirir.

Uygulama

a) Soyadı "Deniz" olan kayıtların Ad verisini "Derya" olarak değiştiren,
b) Soyadı "Şirin" olan kayıtların Soyad verisini "Derya" olarak değiştiren,
c) Adı "Simge" olan kayıtların Soyad verisini "Ayyıldız" olarak değiştiren SQL deyimlerini yazınız.

3.4.4. DELETE Deyimi

Tablodan kayıt silmek amacıyla kullanılır.

```
DELETE FROM tablo ismi [WHERE arama koşulu]
```

Bu deyim WHERE koşulu olmadan kullanılırsa tablodaki tüm kayıtlar silinir.

```
DELETE FROM Kisiler WHERE Ad="Nilsu"
```

Uygulama

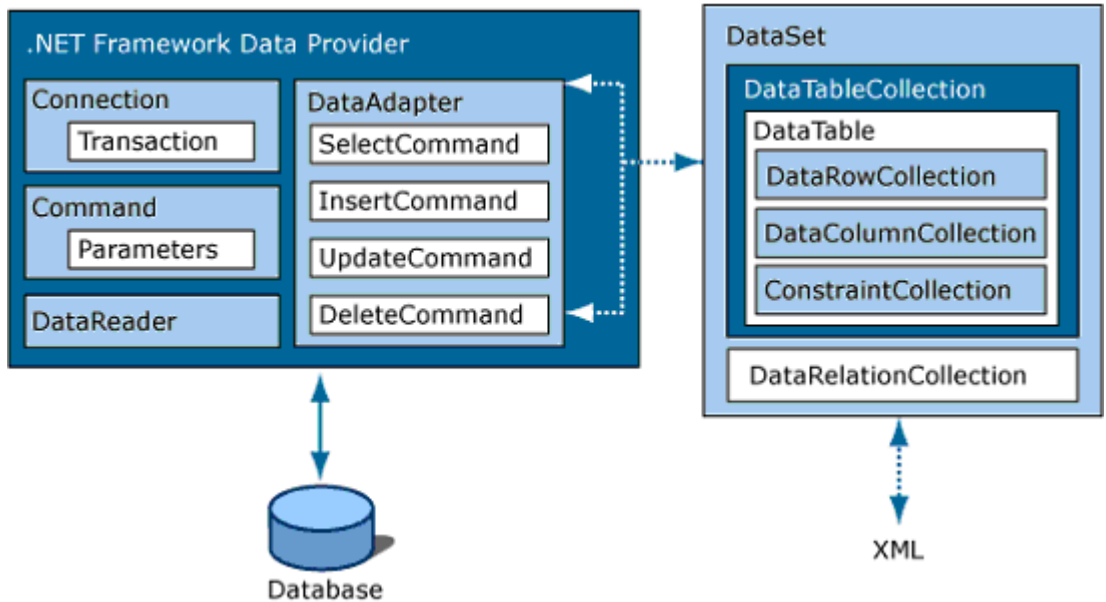
a) Soyadı "Deniz" olan kayıtları silen,
b) Soyadı "Şirin" olan kayıtları silen,
c) Adı "Simge" olan kayıtları silen SQL deyimlerini yazınız.

3.5. ADO.NET ve Veritabanları

ADO.NET nesnelerini kullanarak ASP.NET uygulamalarınız içinde verilerinizi tuttuğunuz veritabanının bir benzerini oluşturabilirsiniz. Böylelikle veritabanındaki verileri uygulama içindeki veritabanının benzeri olan yapıya aktarabilir, üzerinde istediğiniz değişiklikleri yapıp ardından veritabanını güncelleyebilirsiniz.

.NET veritabanlarında tutulan verinin görüntülenmesi için çeşitli kontroller (GridView, Formview, DetailsView, DataList, DataGrid, Repeater vs.) vardır. .NET sahip olduğu nesnelere uygulamaya veri kaynakları arasındaki ilişkiyi yönetir. Uygulamada verileri kullanmak için önce veritabanıyla bağlantı gerçekleştirilir, ardından uygun komutlar kullanılarak veri üzerinde işlemler yapılır.

Veriye erişmek ve değiştirmek için ADO.NET iki bileşene sahiptir: **.NET Framework Data Providers (.NET Veri Sağlayıcıları)** ve **DataSet (Veri Seti)**. Aşağıda .NET SDK dokümanlarından alınan şekil .NET Framework Data Providers ile DataSet arasındaki ilişkiyi göstermektedir.



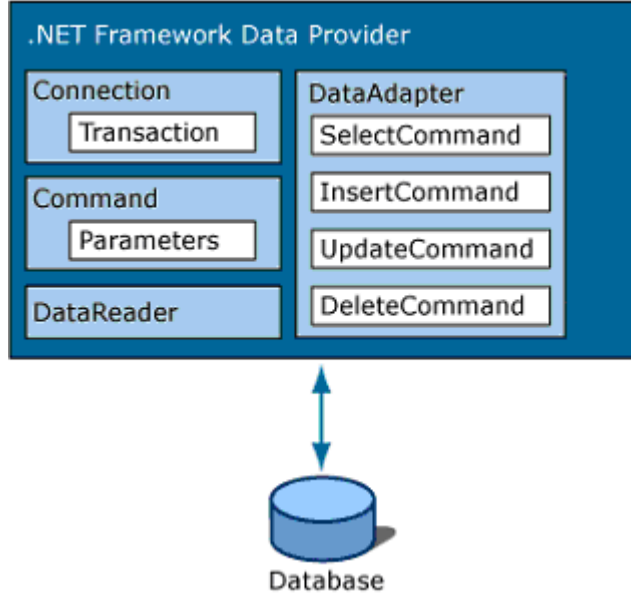
Resim 3.4: .NET Framework Data Provider, veri kaynakları ve DataSet arasındaki ilişki

3.5.1. .NET Veri Sağlayıcısı (.NET Framework Data Provider)

.NET veri sağlayıcı, veri elde etmek, eklemek, güncelleştirmek, silmek için veri kaynağıyla bağlantıya geçmeyi sağlar. .NET'te, çeşitli veri sağlayıcıları vardır. Örneğin, ODBC veri kaynaklarına ulaşmak için kullanılan **ODBC .NET Framework Data Provider**, OLE DB veri kaynaklarına ulaşmak için kullanılan **OLE DB .NET Framework**

Data Provider, SQL Server veri kaynaklarına ulaşmak için kullanılan **SQL Server .NET Framework Data Provider**.

Access veritabanlarına ulaşmak için çoğunlukla **OLE DB .NET Data Provider** kullanılır. Örnek uygulamalarda Access veritabanına erişim sağlayan OLE DB .NET Data Provider kullanılacaktır. OLE DB .NET Data Provider sınıfları **System.Data.OleDb** ad alanında yer almaktadır.



Resim 3.5: Data Provider'ın yapısı ve veritabanıyla ilişkisi

Connection (bağlantı) nesnesi, veritabanına bağlanmak, bir bağlantının özelliklerini almak ve bağlantıyla ilgili olayları işlemek için kullanılır. Veritabanındaki verilere erişmek için öncelikle bir bağlantı oluşturmak gereklidir. Bağlanılacak veritabanının türüne ve uygulamanın gereksinimine göre farklı connection nesnesi kullanılır. Örneğin **ODBCConnection**, **SqlConnection** ve **OleDbConnection**.

Benzer durum aşağıda açıklanacak olan **Command**, **DataReader**, **DataAdapter** nesnelere için de geçerlidir. Yani bir DataAdapter nesnesi kullanılacaksa ihtiyaca göre **ODBCDataAdapter**, **SqlDataAdapter** ve **OleDbDataAdapter** nesnelereinden biri seçilir.

Command (komut) nesnesi, veritabanıyla bağlantı kurulup kullanıma açıldıktan sonra SQL ifadelerini veritabanına karşı yürütmek amacıyla kullanılır. ASP.NET uygulaması bu nesneyi kullanarak veritabanı üzerinde dört temel işlem yapar: Veri elde etme, ekleme, güncelleme ve silme.

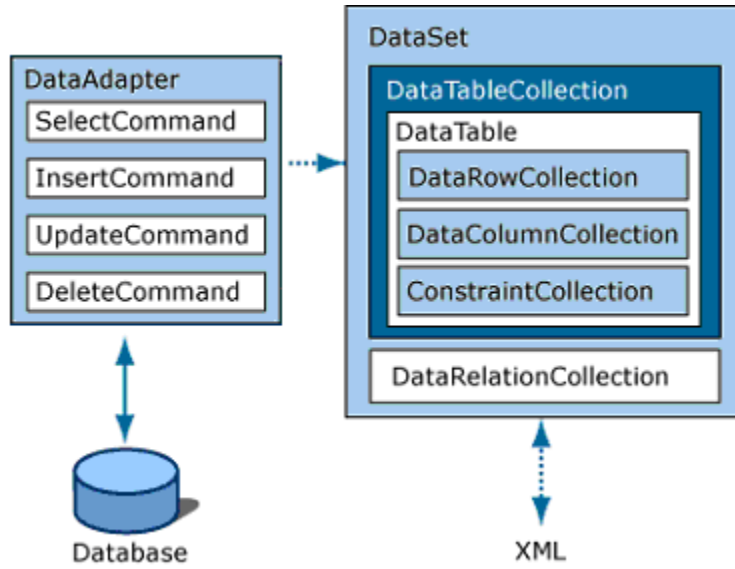
DataReader (veri okuyucu) nesnesi, veritabanından verileri okumak için kullanılır. Bu nesneyle veriler sadece okunabilir olarak elde edildiğinden veritabanındaki veriler üzerinde değişiklik yapmak mümkün değildir.

DataAdapter (veri adaptörü) nesnesi, DataSet nesnesi ile veri kaynağı arasında köprü vazifesi görür. Bir başka deyişle veri kaynağı ile uygulama arasındaki ilişki DataAdapter tarafından sağlanır. DataAdapter nesnesi veri kaynağında SQL komutlarını çalıştırmak için Command nesnesini kullanır. SQL komutlarıyla hem DataSet'e veri yükler hem de DataSet içindeki veri üzerinde yapılan değişiklikleri veritabanına yansıtır.

3.5.2. Veri Seti (DataSet)

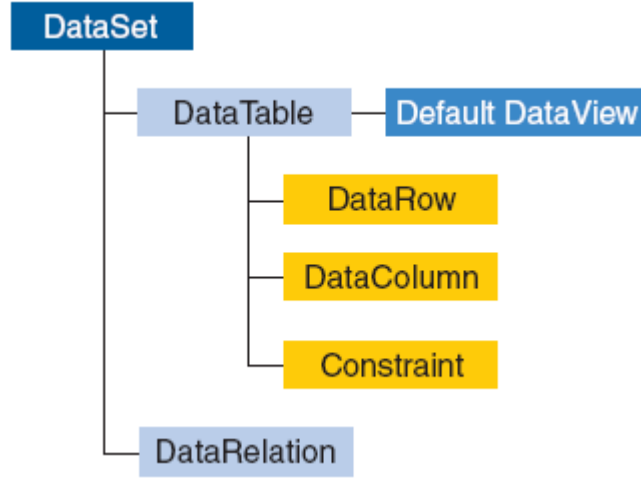
Bir veya daha fazla veri tablosundan (**DataTable**) oluşan, verilerin yapılandırıldığı nesnedir. DataSet, veri kaynağından bağımsız olarak uygulama içinde bir veritabanı oluşturmaya yarar. Kullanılan veri kaynağındakine benzer yapı DataSet içinde oluşturulur. DataSet hafızada tutulur.

DataSet, veri tablosu koleksiyonu (**DataTableCollection**) ve tablolar arasındaki ilişkilerin bir koleksiyonunu (**DataRelationCollection**) içerir. Tablo koleksiyonu veri tablolarından (**DataTable**) oluşur. Tablolar ise satır koleksiyonu (**DataRowCollection**), sütun koleksiyonu (**DataColumnCollection**) ve kısıtlama koleksiyonundan (**constraint collection**) oluşur.



Resim 3.6: DataSet, DataAdapter ve veritabanı arasındaki ilişki

DataSet, DataTable ve DataRelation nesnelerini barındırır. DataTable nesnesi **DataRow**, **DataColumn**, **Constraint** nesnelerinden oluşur. DataTable içindeki verilerin varsayılan bir veri görünümü (**Default DataView**) vardır.



Resim 3.7 :DataSet'in yapısı

DataSet'in kullanımı şu şekildedir: Veritabanındaki veriler DataSet'e alındıktan sonra veritabanıyla kurulan bağlantı kapatılır. Böylelikle veritabanıyla sürekli bağlantı hâlinde kalınmamış olunur. Ardından DataSet'teki veri yapısı üzerinde istenilen değişiklikler yapılır ve üzerinde değişiklik yapılmış veri, tekrar veritabanına aktarılır.

Veritabanından alınarak DataSet içine aktarılan veri web sunucunun hafızasında saklanır. Sunucu hafızasına fazla yüklenmemek için tüm veriler değil sadece gerekli olan veriler uygulamaya aktarılmalıdır.

DataSet içindeki veriler XML dosyasından, kullanıcının girdiği verilerden, metin dosyasından veya veritabanı programından elde edilebilir.

DataSet sınıfı **System.Data** ad alanında bulunmaktadır. DataSet iki yöntemle oluşturulup kullanılabilir.

Birinci yöntem, DataTable, DataRow gibi tüm veritabanı yapılarıyla programatik olarak boş bir DataSet oluşturmaktır. İkinci yöntem, bir veritabanı ile temasa geçip veritabanındaki yapıyı olduğu gibi DataSet'e aktarmaktır. Bu modülde ikinci yöntem kullanılacaktır.

OleDbDataAdapter kullanarak Access veritabanından DataSet'e veri aktarma işlemi aşağıdaki şekilde gerçekleşir:

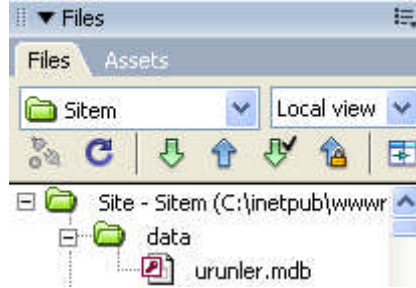
1. İstemci, bir ASP.NET sayfası çağırarak web sunucusundan (örneğin, IIS) istekte bulunur.
2. Sayfa, OleDbConnection ve OleDbDataAdapter nesnelerini kullanıp, veritabanı ile bağlantıya geçerek verileri DataSet'a aktarır. DataSet'teki veriler, uygulamayı yürüten sunucunun hafızasında geçici bir süre muhafaza edilir. DataSet içindeki

veriler veri bağlanılabilen web sunucu kontrolleri (GridView vs.) kullanılarak istemci bilgisayarda görüntülenir.

3. DataSet'e veri geldikten ve istemciye yollandıktan sonra hem istemci ile web sunucusu hem de web sunucusu ile veritabanı arasındaki bağlantı kesilir. Yapı, **bağlı olmayan (disconnected) ortam** olarak çalışır.
4. Veri üzerinde değişiklikler yapıldıktan sonra istemci, veriyi web sunucusu üzerindeki DataSet'e gönderir. Veri, DataAdapter kullanılarak veritabanına aktarılır.

Veritabanına bağlanarak verilerin OleDbDataAdapter nesnesi yardımıyla DataSet'e yüklendiği ve DataSet'ten alınarak DataGrid kontrolünde görüntülediği örnek bir sayfayı tamamen kod kullanarak oluşturalım. Sayfadan bağlantı kurulacak **urunler.mdb** veritabanı dosyası **sitem** isimli web site klasöründe oluşturulan **data** klasörü içinde tutulmaktadır.

NOT: DataGrid ASP.NET 1.x sürümlerinde kullanılan bir veri bağlanılan (**data bound**) kontroldür. ASP.NET 2.0'da DataGrid'den daha üstün özelliklere sahip GridView kontrolü geliştirilmiştir.



Resim 3.8: Files paneli

Örnek **urunler.mdb** dosyası Access 2002 sürümünde oluşturulmuştur. İçinde **Kitaplar** isimli bir tablo bulunmaktadır.

Kitaplar : Tablo		
	Alan Adı	Veri Türü
?	Nu	Metin
	Kitapadi	Metin
	Yazaradi	Metin
	Yayınevi	Metin

Resim 3.9: Kitaplar tablosu tasarım görünümü

Kitaplar : Tablo				
	Nu	Kitapadi	Yazaradi	Yayinevi
	1	Nutuk	M.Kemal Atatürk	Kanarya
	2	Mesnevi'den Seçmeler	Mevlana	Serçe
	3	Çalikuşu	Reşat Nuri Güntekin	Doğan
	4	Dostlar Beni Hatırlasın	Aşık Veysel	Kartal
	5	Yaşar Ne Yaşar Ne Yaşamaz	Aziz Nesin	Aslan
	6	Memleketimden İnsan Manzaraları	Nazım Hikmet	Kaplan
	7	Türkçe'nin Sırları	Nihad Sami Banarlı	Jaguar
	8	Babalar ve Oğullar	Turgenyev	Karga

Resim 3.10: Kitaplar tablosu veri sayfası görünümü

vtErisim.aspx

```

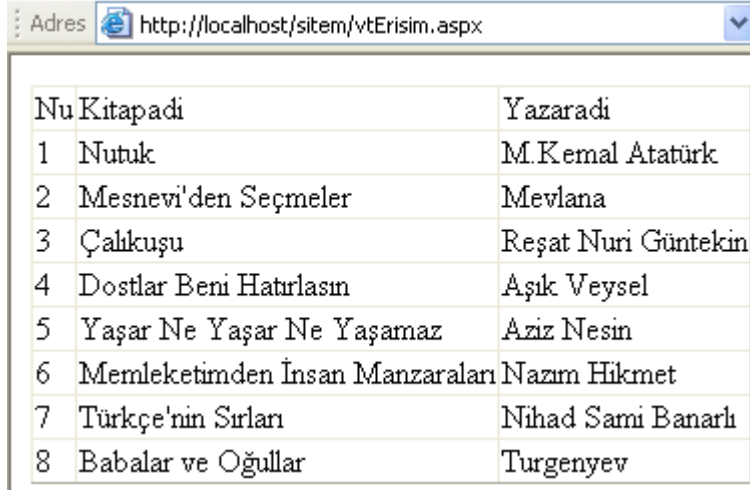
1 <%@ Page Language="VB" Debug="true" ContentType="text/html" ResponseEnc
2 <%@ Import Namespace="System.Data" %>
3 <%@ Import Namespace="System.Data.OleDb" %>
4 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://t
5 <html>
6 <head>
7 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
8 <title>Veri Erişimi Örneği</title>
9 <script runat="server">
10 Sub Page_Load(Src As Object, E As EventArgs)
11     'Connection nesnesini tanımla.
12     Dim baglanti As New OleDbConnection
13     'Kullanılacak provider ve veri tabanı yolunu belirle.
14     baglanti.ConnectionString = "Provider = Microsoft.Jet.OLEDB.4.0;" & _
15     "data source=" & Server.MapPath("data\urunler.mdb")
16     'Bağlantıyı aç.
17     baglanti.open
18     'Command nesnesini tanımla.
19     Dim komut As New OleDbCommand
20     'Command nesnesini Connection nesnesiyle ilişkilendir.
21     komut.connection = baglanti
22     'Command nesnesinin SQL komutunu belirle.
23     komut.CommandText = "SELECT * FROM kitaplar"
24     'Adapter nesnesini tanımla.
25     Dim adaptor As New OleDbDataAdapter
26     'Adapter nesnesini Command nesnesiyle ilişkilendir.
27     adaptor.SelectCommand = komut
28     'DataSet nesnesini tanımla.

```

```

29 Dim dsurunler As New DataSet
30 'Verileri DataSet içindeki tabloya doldur.
31 adaptor.Fill(dsUrunler,"tkitaplar")
32 'DataGrid'in veri kaynağı olarak DataSet'i belirle.
33 datagrid1.DataSource = dsurunler
34 'DataGrid'e veriyi bağla.
35 datagrid1.DataBind()
36 'Bağlantıyı kapat
37 baglanti.close
38 End Sub
39 </script>
40 </head>
41 <body>
42 <form runat="server" name="form1" method="post" action="">
43 <asp:datagrid ID="datagrid1" runat="server"></asp:datagrid>
44 </form>
45 </body>
46 </html>

```



Nu	Kitapadi	Yazaradi
1	Nutuk	M.Kemal Atatürk
2	Mesnevi'den Seçmeler	Mevlana
3	Çalkıuşu	Reşat Nuri Güntekin
4	Dostlar Beni Hatırlasın	Aşık Veysel
5	Yaşar Ne Yaşar Ne Yaşamaz	Aziz Nesin
6	Memleketimden İnsan Manzaraları	Nazım Hikmet
7	Türkçe'nin Sırları	Nihad Sami Banarlı
8	Babalar ve Oğullar	Turgenyev

Resim 3.11: vtErisim.aspx sayfasının tarayıcıdaki görüntüsü

1-3. satırlarda hata mesajlarını görmek için **Page** bildirimine **Debug="true"** ifadesi eklenmiştir. Her seferinde veri nesnelерinin tam ismini yazmak zorunda kalmamak için gerekli ad alanları sayfaya import edilmiştir.

12. satırda **baglanti** isimli yeni bir **OleDbConnection** nesnesi tanımlanmıştır.

14. ve 15. satırlarda **OleDbConnection** nesnesinin **ConnectionString** özelliği kullanılarak **provider** ve veritabanının yolu belirlenmiştir. Access veritabanına bağlantı yapılacağından sürücü stringi (**driver string**) olarak **Microsoft.Jet.OLEDB.4.0** kullanılmıştır. Farklı bir veritabanına bağlanılsaydı sürücü stringi de farklı olacaktı. Örneğin,

Microsoft SQL Server kullanılırsa **SQLOLEDB** stringi kullanılacaktır. **Server.MapPath** ile veritabanının sunucu üzerindeki yeri belirtilmiştir. Test amaçlı olduğunda aşağıdaki gibi dosyanın doğrudan yolunu da yazabilirsiniz.

```
baglanti.ConnectionString = "Provider = Microsoft.Jet.OLEDB.4.0;" & _  
"data source=C:\Inetpub\wwwroot\Sitem\data\urunler.mdb"
```

17. satırda bağlantı açılmıştır. **Baglanti** isimli **OleDbConnection** nesnesi **Open** metodu kullanılarak aktif hale getirilmiştir. Farklı bir veritabanına bağlantı gerçekleştirilseydi ona uygun **Connection** nesnesi seçilecekti. Örneğin, SQL Server veritabanlarına bağlantı için **SqlConnection** nesnesi kullanılacaktı.

19. satırda **komut** isimli yeni bir **OleDbCommand** nesnesi tanımlanmıştır. 21. satırda **OleDbCommand** nesnesi ile **OleDbConnection** nesnesi **Connection** özelliği ile ilişkilendirilmiştir. 23. satırda **OleDbCommand** nesnesinin **CommandText** özelliğine SQL komutu atanmıştır. Bu komut ileriki satırlarda kullanılarak **urunler.mdb** veritabanındaki **kitaplar** tablosundaki tüm kayıtlar elde edilecektir.

25. satırda **adaptor** isimli yeni bir **OleDbAdapter** nesnesi tanımlanmıştır. 27. satırda **OleDbAdapter** nesnesinin **SelectCommand** özelliği ile **OleDbCommand** nesnesi ilişkilendirilmiştir.

29. satırda **dsurunler** isimli yeni bir **DataSet** nesnesi tanımlanmıştır. 31. satırda **OleDbAdapter** nesnesinin **Fill** metodu ile **CommandText** ile seçilen veriler **dsurunler DataSet**'inde tanımlanan **kitaplar** tablosuna doldurulmuştur. Bu tablo artık ASP.NET uygulamasının bir parçasıdır. İstenirse bu tablodaki veriler üzerinde istenilen değişiklikler yapılabilir ve bu değişiklikler veritabanına aktarılabilir.

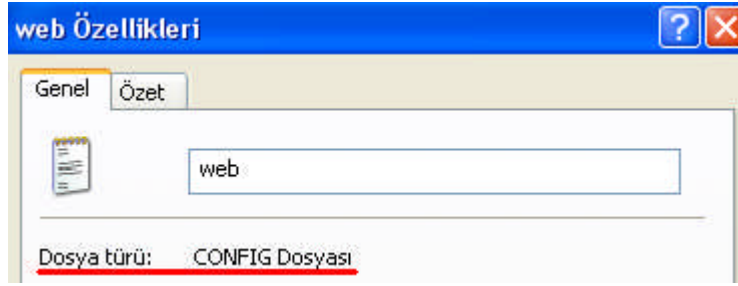
33. satırda **Datagrid1** isimli **DataGrid** kontrolünün **DataSource** özelliğine **DataSet** atanmıştır. 35. satırda **Datagrid**'e **DataBind** metoduyla veri bağlanmıştır. 33. ve 35. satırlar sayesinde **DataSet** içinde bulunan veriler **Datagrid** kontrolüne aktarılmıştır.

37. satırda **OleDbConnection** nesnesi **Close** metodu ile kapatılmıştır.

NOT: Bu yöntemde **Open** ve **Close** metotları kullanılmasa da olur. Veritabanı bağlantısını açma ve kapama işlemleri **DataAdapter** tarafından yapılmaktadır.

Bağlantı stringini (**ConnectionString**) belirtmenin diğer bir yolu **web.config** yapılandırma dosyasını kullanmaktır. Bu yolu kullanarak **vtErisimConfig.aspx** sayfasını hazırlayalım.

Sitem web site klasörü içerisinde bir metin belgesi oluşturup içine aşağıdakileri yazarak **web.config** ad ve uzantısıyla kaydediniz. Dosyanızın üzerinde sağ fare tuşuna bastığınızda gelen kısayol menüsünden **özellikler** komutunu verdiğinizde dosyanın türünün **CONFIG** dosyası olduğu görülür.



Resim 3.12: Web.config yapılandırma dosyası

Bu dosya içerisine aşağıdaki satırları yazınız.

```
<configuration>
  <appSettings>
    <add key="vtbaglanti"
      value="Provider=Microsoft.Jet.OLEDB.4.0;
      Data Source=C:\Inetpub\wwwroot\Sitem\data\urunler.mdb" />
  </appSettings>
</configuration>
```

Web.config dosyasını ayarladıktan sonra vtErisim.aspx dosyasındaki 12-15. satırları silip yerine aşağıdaki kodları yazarak dosyayı **vtErisimConfig.aspx** adıyla kaydediniz.

```
Dim baglanti As New OleDbConnection _
(ConfigurationSettings.AppSettings("vtbaglanti"))
```

Değişiklik yapılan kısım şu şekilde olacaktır:

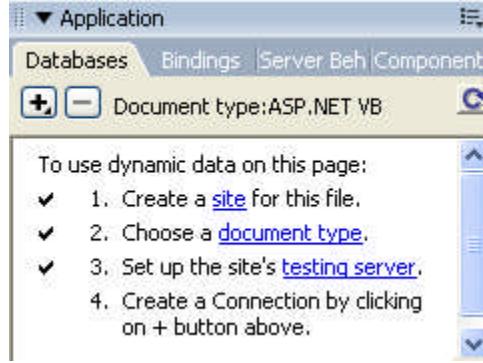
```
11 'Connection nesnesini tanımla.
12 'Kullanılacak provider ve veri tabanı yolunu belirle.
13 Dim baglanti As New OleDbConnection _
14 (ConfigurationSettings.AppSettings("vtbaglanti"))
15 'Bağlantıyı aç.
```

Yukarıdaki satır ile, yapılandırma dosyası içinde tanımlanan anahtar (**key**) kullanan **baglanti** isimli **OleDbConnection** nesnesi tanımlanmaktadır. Web.config dosyasında bağlantı satırını belirleyip kullanmak özellikle çok sayfanın aynı bağlantı stringini kullandığı durumlarda yararlıdır. Bağlantı satırını değiştirmek gerektiğinde tüm sayfaları teker teker değiştirmek zorunluluğunu ortadan kaldırır. Dreamweaver tarafından kullanılan yöntem de budur. Dreamweaver web.config dosyası varsa dosyaya anahtar ekler, dosya yoksa oluşturur.

VtErisim.aspx ve vtErisimConfig.aspx sayfalarında .NET Framework içerisindeki DataSet ve DataGrid (asp:DataGrid) nesneleri kullanılmıştır. ASP.NET'te istenirse Dreamweaver kontrolleri de kullanılabilir. Örneğin, <MM:DataSet.....></MM:DataSet>

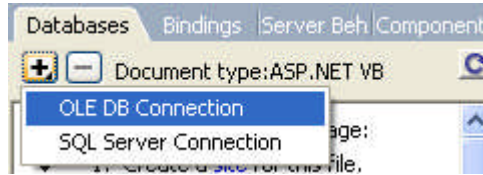
Şimdi de Dreamweaver kontrollerini kullanarak **vtErisimDW.aspx** isminde bir sayfa oluşturalım.

Bilindiği gibi sayfadan bir veritabanına erişmek için bağlantı ifadesinin **Provider** parametresinde uygun sürücünün belirtilmesi gereklidir. Bu işlem **Application** panelinden yapılabilir.



Resim 3.13: Application paneli Databases sekmesi

Panelde sayfada dinamik data kullanmak için (**To use dynamic data on this page**) yapılması gereken dört işlem basamağı listelenmiştir. Bu dosya için bir site oluşturma (**Create a site for this file.**), belge tipi seçme (**Choose a document type**), sitenin test sunucusunu ayarlama (**Set up the site's testing server**) işlemlerinin tamamlandığı onay işareti ile belirtilmiştir. Dördüncü ve son işlem yukarıdaki + düğmesini tıklatarak bağlantı oluşturmaktır (**Create a Connection by clicking on + button above**). İlgili düğmeye basıp bağlantıyı oluşturmaya başlayalım.



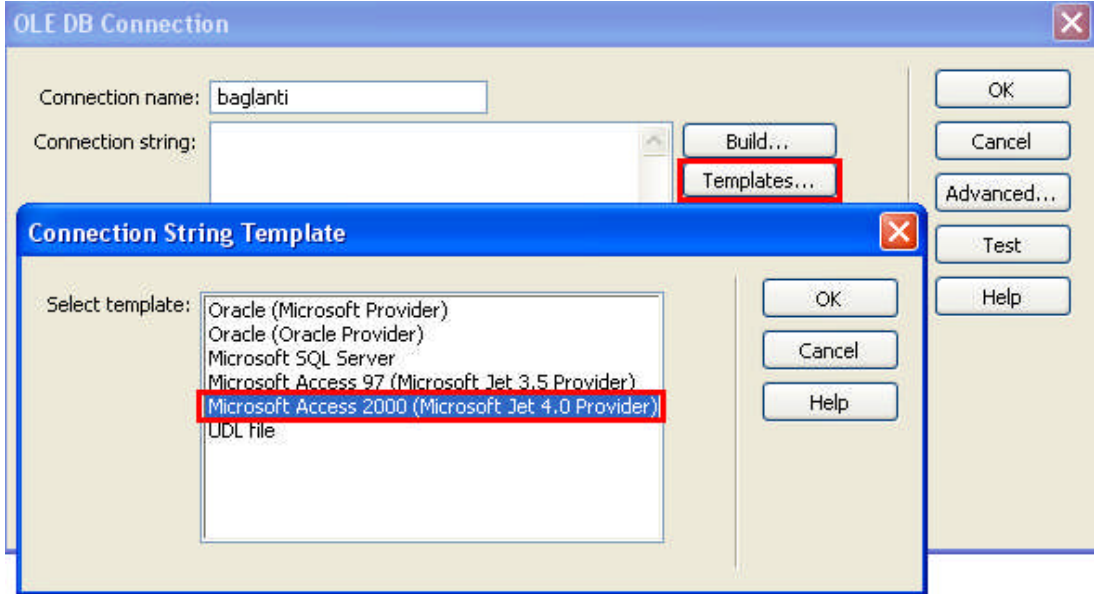
Resim 3.14: OLE DB Connection oluşturma

OLE DB Connection penceresinde **Connection name** kutusuna **baglanti** yazınız. **Connection string** oluşturmak için **Templates** düğmesini tıklatıp gelen pencerede **Microsoft Access 2000 (Microsoft Jet 4.0 Provider)** şablonunu seçip **OK** düğmesini tıklatınız (Resim 3.15).

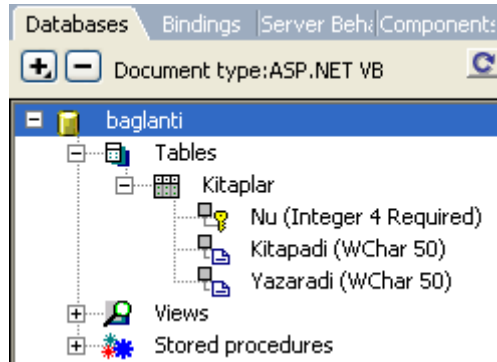
Connection string kutusunun içeriğini aşağıdaki şekilde değiştiriniz. Ardından **OK** düğmesini tıklatınız.

```
"Provider=Microsoft.Jet.OLEDB.4.0;  
Data Source=C:\inetpub\wwwroot\Sitem\data\urunler.mdb"
```

Application panelinin **Databases** sekmesinde **baglanti** isimli **Connection**'ı alt seçeneklerini açtığınızda Resim 3.16'daki görüntüyle karşılaşacaksınız.



Resim 3.15: Connection String Template



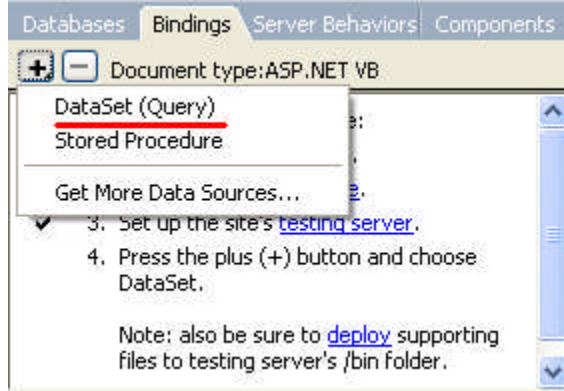
Resim 3.16: Baglanti isimli Connection

Bu aşamada oluşturulan **web.config** dosyasının içeriği aşağıdaki gibidir.

```
<configuration>
  <appSettings>
    <add key="MM_CONNECTION_HANDLER_baglanti" value="default_oledb.htm" />
    <add key="MM_CONNECTION_STRING_baglanti" value="Provider=Microsoft.Jet.OLEDB.4.0;
Data Source=C:\inetpub\wwwroot\Sitem\data\urunler.mdb" />
    <add key="MM_CONNECTION_DATABASETYPE_baglanti" value="OleDb" />
    <add key="MM_CONNECTION_SCHEMA_baglanti" value="" />
    <add key="MM_CONNECTION_CATALOG_baglanti" value="" />
  </appSettings>
</configuration>
```

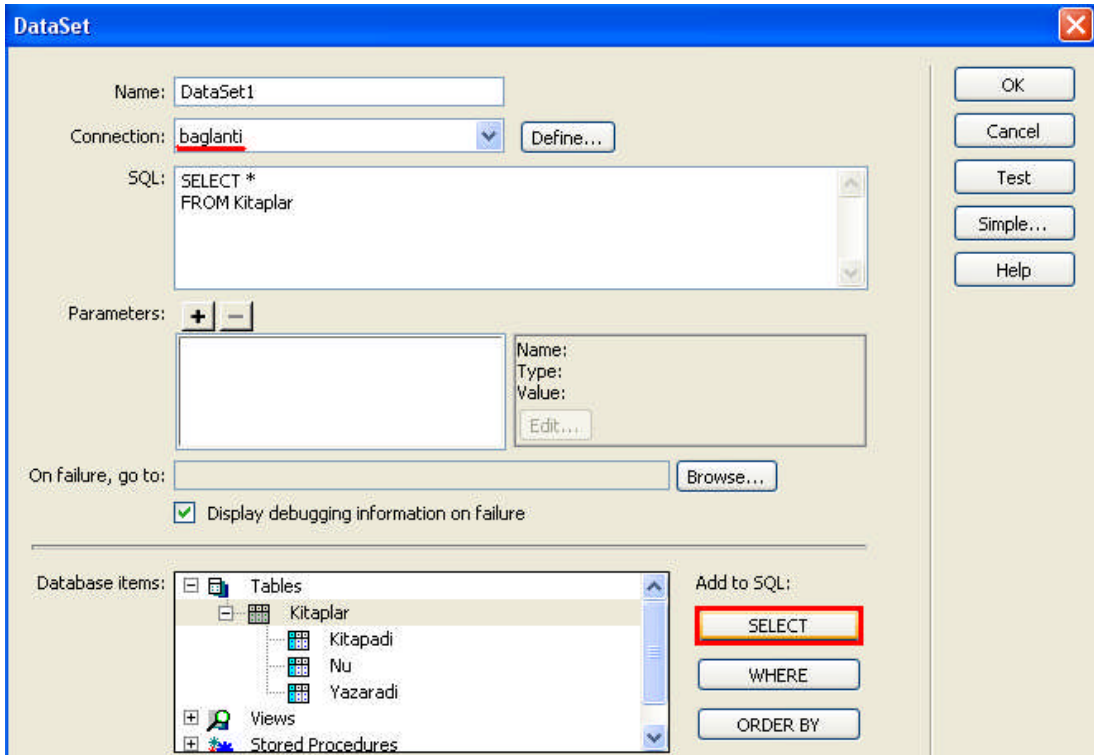
Artık bir DataSet veya bir veri bağlanabilecek kontrol eklediğinizde yukarıda oluşturduğunuz bağlantıya başvurabilirsiniz.

Bir DataSet oluşturarak sayfayı geliştirmeye devam edelim. **Application** panelinin **Bindings** sekmesindeki + düğmesini tıklatınız. Ardından **DataSet (Query)** komutunu veriniz.



Resim 3.17: DataSet Oluşturma

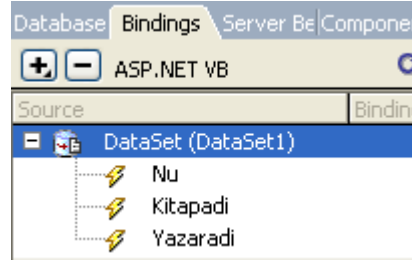
Gelen pencereden **Advanced** düğmesini tıklatınız. **Connection** kutusundan **baglanti** isimli **Connection**'ı seçiniz. **Database Items** kısmından **Kitaplar** tablosunu ardından **SELECT** düğmesini tıklatınız.



Resim 3.18: DataSet'in Advanced penceresi

Select düğmesini tıkladıktan sonra **SQL** kutusuna uygun SQL deyimini yazılmıştır. İsterseniz doğrudan uygun SQL deyimini buraya yazabilirsiniz. Ayarları onaylamadan önce **Test** düğmesini tıklararak bağlantıyı test ediniz. **Ok** düğmesini tıklarınız.

Bindings sekmesinde veri kaynağı olarak kullanıma hazır hâlde DataSet nesnesi görüntülenecektir.



Resim 3.19: DataSet

Şimdi de sayfaya eklenen kodları inceleyelim. Page bildiriminin altına aşağıdaki kodlar eklenmiştir.

```
<%@ Register TagPrefix="MM" Namespace="DreamweaverCtrls"
Assembly="DreamweaverCtrls,version=1.0.0.0,publicKeyToken=
836f606ede05d46a,culture=neutral" %>
```

Bu satırlar ile Macromedia'nın özel MM: kontrolleri sayfaya kaydedilmektedir (**register**). Sitenizin kök klasöründeki **bin** klasöründe bulunan ve MM: kontrollerini barındıran DreamweaverCtrls.dll assembly olarak belirtilmektedir.

@Register bildiriminin altında da aşağıdaki kodlar yer almaktadır.

```
<MM:DataSet id="DataSet1" runat="Server" IsStoredProcedure="false"
ConnectionString='<%#
System.Configuration.ConfigurationSettings.AppSettings
("MM_CONNECTION_STRING_baglanti") %>'
DatabaseType='<%#
System.Configuration.ConfigurationSettings.AppSettings
("MM_CONNECTION_DATABASETYPE_baglanti") %>'
CommandText='<%# "SELECT * FROM Kitaplar" %>'
Debug="true">
</MM:DataSet>
```

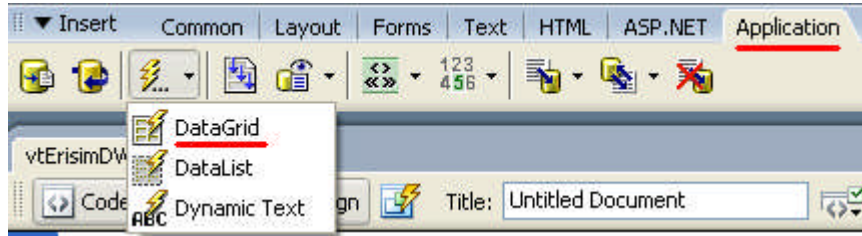
Koyu yazılmış satırlar uygulamanızın web.config dosyasından aldığı değerleri göstermektedir. Web.config dosyası daha önceden oluşturulmamışsa Dreamweaver otomatik olarak oluşturacaktır. **CommandText** ise SQL deyimini içermektedir.

Aşağıda satır ise sayfa çağrıldığında sayfadaki tüm veriyle ilişkili kontrollere verinin bağlanmasını sağlar.

```
<MM:PageBind runat="server" PostBackBind="true" />
```

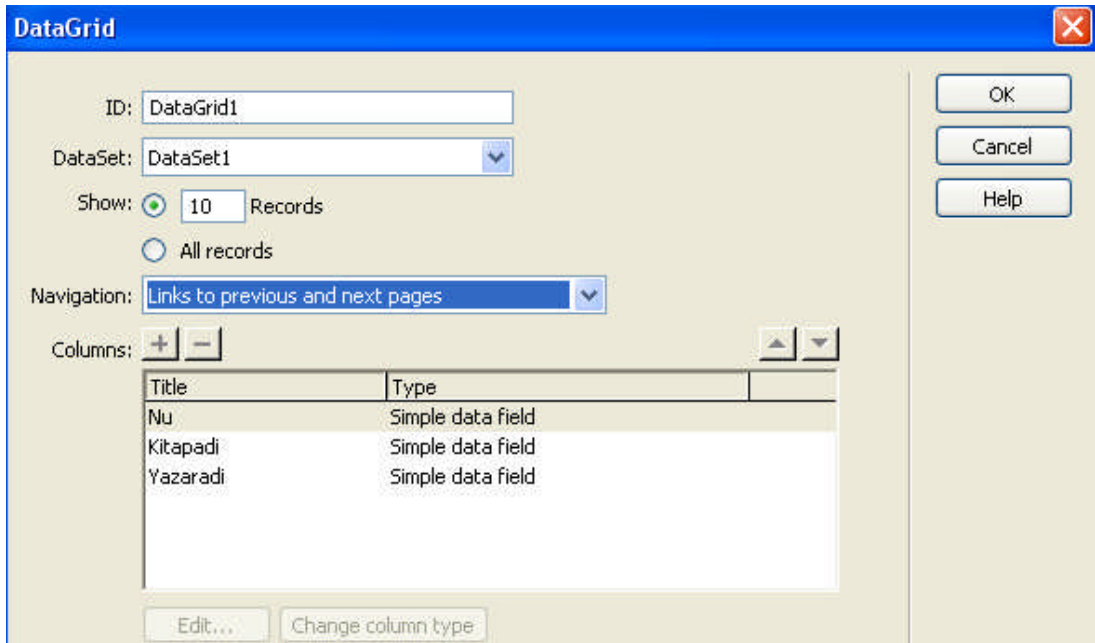
Şimdi mevcut veriyi sayfa üzerinde görüntüleyelim. Verileri kontrollere bağlayalım.

İmleci **body** etiketleri arasına yerleştikten sonra **Insert** çubuğunun **Application** sekmesinde **Dynamic Data** simgesinden **DataGrid**'i seçiniz.



Resim 3.20: DataGrid ekleme

DataGrid penceresinde **DataSet** kısmından **DataSet1**'i seçiniz. **Show** kısmından **All Records** ile tüm kayıtların görüntülenmesi, kutuya girilecek sayıyla ise bir sayfada belirtilen sayıda kaydın görüntülenmesi sağlanır. **Navigation** penceresinde ise önceki ve sonraki sayfalara link verilmesi (**Links to previous and next pages**) veya her sayfaya link verilmesi (**Numbered links to every page**) seçenekleri mevcuttur. Aşağıdaki ayarları yaparak **OK** düğmesini tıklatınız.



Resim 3.21: DataGrid penceresi

Artık verileriniz DataGrid kontrolü ile sayfanızda görüntülenmektedir.



Nu	Kitapadi	Yazaradi
1	Nutuk	M.Kemal Atatürk
2	Mesnevi'den Seçmeler	Mevlana
3	Çalığışu	Reşat Nuri Güntekin
4	Dostlar Beni Hatırlasın	Aşık Veysel
5	Yaşar Ne Yaşar Ne Yaşamaz	Aziz Nesin
6	Memleketimden İnsan Manzaraları	Nazım Hikmet
7	Türkçe'nin Sırları	Nihad Sami Banarlı
8	Babalar ve Oğullar	Turgenyev

Resim 3.22: vtErisimDW.aspx sayfasının tarayıcıdaki görüntüsü

```
<asp:DataGrid id="DataGrid1" runat="server"
AllowSorting="False"
AutoGenerateColumns="false"
CellPadding="3"
CellSpacing="0"
ShowFooter="false"
ShowHeader="true"
DataSource="<%# DataSet1.DefaultView %>"
PagerStyle-Mode="NextPrev"
AllowPaging="true"
AllowCustomPaging="true"
PageSize="<%# DataSet1.PageSize %>"
VirtualItemCount="<%# DataSet1.RecordCount %>"
OnPageIndexChanged="DataSet1.OnDataGridPageIndexChanged" >
```

AllowSorting="False", sıralama özelliğinin etkin hâle getirilmemiştir.

AutoGenerateColumns="False" ile otomatik sütun oluşturma özelliği pasif olarak ayarlanmıştır. Sütunlar **Columns** kısmında **<asp:BoundColumn>** etiketleriyle oluşturulacaktır.

DataSource özelliğine **DataSet**'in varsayılan görünümü (**default view**) atanmıştır. **PagerStyle-Mode** özelliği ileri-geri (**Next-previous**) linklerini gösterecek şekilde ayarlanmıştır.

OnPageIndexChanged özelliği **DreamweaverCtrls.dll** içinde bulunan ve bir sayfadan diğer sayfaya geçişi kontrol eden olay işleme programını belirtmektedir. Bu olay işleme programı **DataGrid**'in aktif sayfasını yeni tıklanan sayfa olarak ayarlamaktadır. Olay işleme programında bulunan aşağıdaki satırla bu işlem gerçekleştirilmektedir.

```
DataGrid1.CurrentPageIndex = e.NewPageIndex
```

AllowPaging özelliğine **true** değeri atanarak sayfalama etkin hâle getirilmiştir.

```
<HeaderStyle HorizontalAlign="center" BackColor="#E8EBFD"
<ItemStyle BackColor="#F2F2F2" Font-Name="Verdana, Arial,
<AlternatingItemStyle BackColor="#E5E5E5" Font-Name="Verd
<FooterStyle HorizontalAlign="center" BackColor="#E8EBFD"
<PagerStyle BackColor="white" Font-Name="Verdana, Arial, :
```

Yukarıdaki satırlarla **DataGrid**'in kullanıcıya nasıl görüneceği ayarlanmıştır.

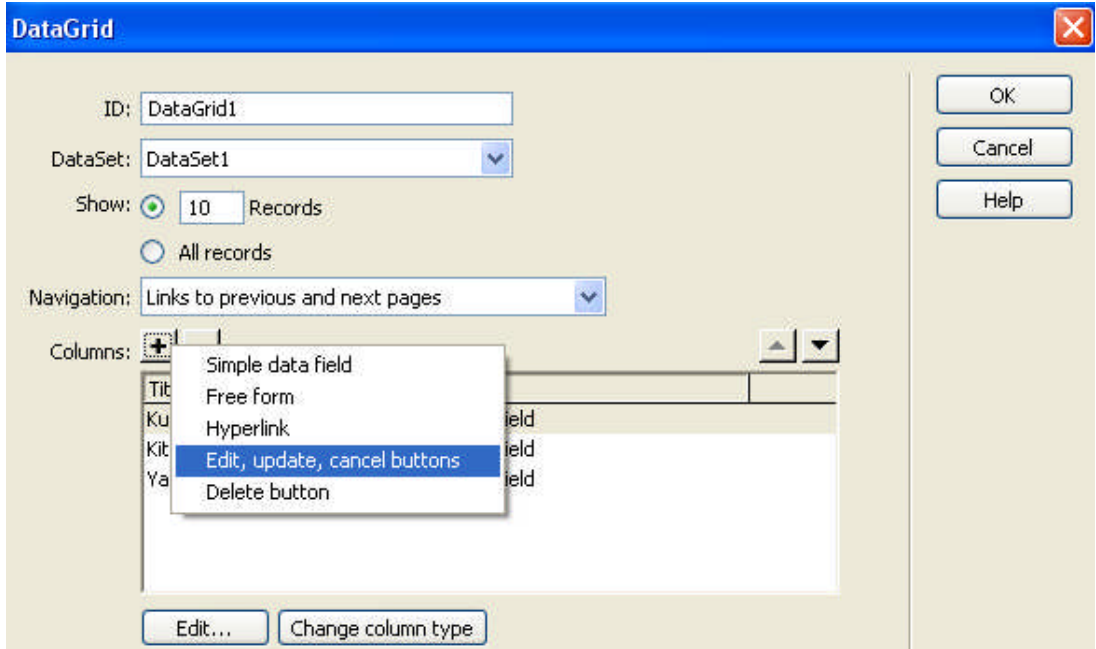
```
<Columns>
<asp:BoundColumn DataField="Nu"
    HeaderText="Nu"
    ReadOnly="true"
    Visible="True"/>
<asp:BoundColumn DataField="Kitapadi"
```

Columns kısmında veri bağlanılan sütunlarla (**BoundColumn**) ilgili bilgiler verilmiştir. **HeaderText**, sütunun başlığını belirtmektedir. **BoundColumn**'lardaki **HeaderText** özelliklerini sırasıyla **Sıra Nu**, **Kitap Adı**, **Yazar Adı** şeklinde değiştirelim. **ReadOnly=true**, bu sütundaki değerlerin sadece okunabilir olduğunu göstermektedir. **ReadOnly** özelliği **False** olarak ayarlandığında düzenleme (**Edit**) düğmeleri tıklandığında bu veri alanının içeriği değiştirilebilir. **Visible=true**, bu sütunun görüntüleneceğini belirtir.

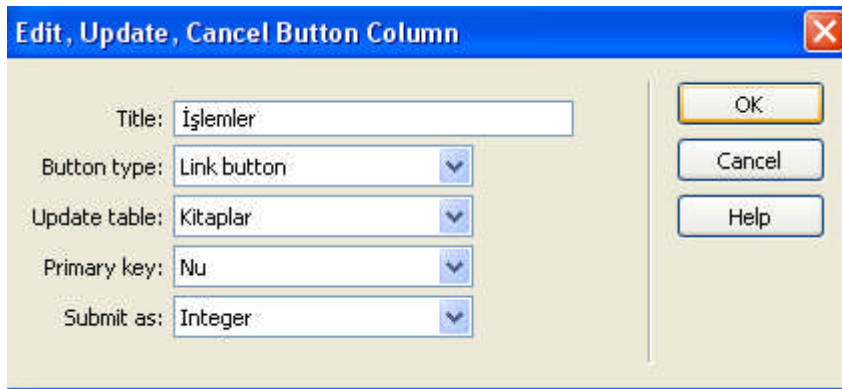
vtErisimDW sayfası üzerinde çeşitli değişiklikler yapalım. **Application** panelinde **Server Behaviors** sekmesindeki **DataGrid**'i çift tıklatarak penceresini açınız. **Columns** kısmından + düğmesini tıklatarak **Edit, update, cancel buttons** komutunu seçiniz (Resim 3.23).

Edit, Update, Cancel Button Column penceresinde Resim 3.24'teki ayarları yapıp **OK** düğmesini tıklatınız. Aşağıdaki satırlar sayfanıza eklenecektir.

```
<asp:EditCommandColumn
    ButtonType="LinkButton"
    CancelText="Cancel"
    EditText="Edit"
    HeaderText="İşlemler"
    UpdateText="Update"
    Visible="True"/>
```



Resim 3.23: Edit, Update, Cancel Butonları ekleme

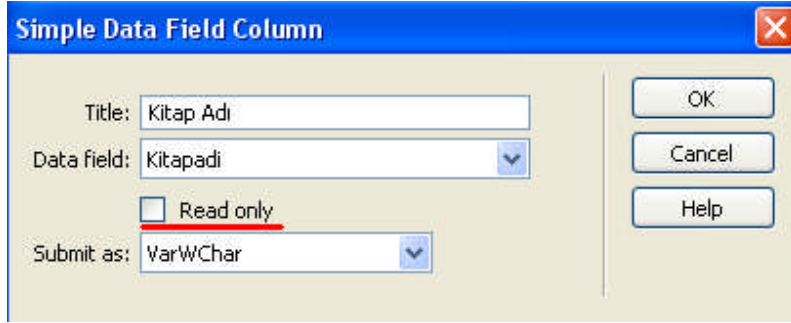


Resim 3.24: Edit, Update, Cancel Button Columns penceresi

EditCommandColumn'ın **CancelText**, **EditText**, **UpdateText** özelliklerindeki **Cancel**, **Edit**, **Update** İngilizce ifadeleri sırasıyla **İptal**, **Düzenle**, **Güncelle** şeklinde değiştiriniz.

LinkButton yerine **PushButton** kullanılmak istenirse **ButtonType** özelliğinde LinkButton yerine PushButton değeri atanmalıdır.

Üzerinde değişiklik yapabilmek için **Kitapadi**, **Yazaradi** veri alanlarının **ReadOnly** özelliklerine **False** değerini atamak gereklidir. **ReadOnly** özelliği **False** yapmak için **DataGrid** penceresinde ilgili veri alanını seçtikten sonra **Edit** düğmesini tıklattınca gelen **Simple Data Field Column** penceresindeki **ReadOnly** onay kutusunun işaretini kaldırınız (Resim 3.25). **Title** metin kutusunu kullanarak sütunun başlığını değiştirebilirsiniz.



Resim 3.25: Simple Data Field Column

Sayfa çağrıldığında görüntüsü aşağıdaki şekilde olacaktır.

Sıra Nu	Kitap Adı	Yazar Adı	İşlemler
1	Nutukb	M.Kemal Atatürk	Düzenle
2	Mesnevi'den Seçmeler	Mevlana	Düzenle
3	Çalığışu	Reşat Nuri Güntekin	Düzenle
4	Dostlar Beni Hatırlasın	Aşık Veysel	Düzenle
5	Yaşar Ne Yaşar Ne Yaşamaz	Aziz Nesin	Düzenle
6	Memleketimden İnsan Manzaraları	Nazım Hikmet	Düzenle
7	Türkçe'nin Sırları	Nihad Sami Banarlı	Düzenle
8	Babalar ve Oğullar	Turgenyev	Düzenle

Resim 3.26: vtErisimDW.aspx sayfasının tarayıcıdaki görüntüsü

Düzenle düğmesi tıklatıldığında veriler metin kutusu içinde düzenlemeye hazır duruma gelecektir.

Sıra Nu	Kitap Adı	Yazar Adı	İşlemler
1	<input type="text" value="Nutuk"/>	<input type="text" value="M.Kemal Atatürk"/>	Güncelle İptal
2	Mesnevi'den Seçmeler	Mevlana	Düzenle

Resim 3.27: Düzenleme modundaki vtErisimDW.aspx sayfası

NOT: Dreamweaver MX2004 programı ASP.NET 1.x sürümüne destek verirken ASP.NET 2.0 sürümüne tam destek vermemektedir.

3.6. Veri Kaynağı Kontrolleri (Data Source Controls)

ADO.NET'te farklı veri kaynaklarına bağlanmak için çeşitli veri kaynağı kontrolleri vardır. ASP.NET 2.0 sürümüyle birlikte gelen bu kontroller **SqlDataSource**, **ObjectDataSource**, **AccessDataSource**, **SiteMapDataSource**, **XmlDataSource** kontrolleridir.

VtErisim.aspx örneğinde olduğu gibi veritabanına bağlanılabilir. Veri kaynağı kontrolleri ile veri kaynaklarına bağlanma ise yeni bir yöntemdir. Bu yöntem kod yazımını azaltmaktadır fakat Dreamweaver'da bu kontrole görsel destek olmadığından bu kontrol ve ASP.NET ile gelen diğer yeni kontroller daha fazla kod yazılarak kullanılmaktadır.

Veri kaynağı kontrolleriyle veriye erişim sağlandıktan sonra bu verilerin net sayfasında görüntülenmesi için veri bağlanılan kontroller (**data-bound controls**) kullanılır. Bu veri kontrolleri, ASP.NET 1.x sürümündeki DataGrid, DataList, Repeater kontrollerine ek olarak ASP.NET 2.0 sürümüyle birlikte gelen GridView, DetailsView, FormView, TreeView kontrolleri olabilir.

Access veritabanlarına bağlanmak için **AccessDataSource** veri kaynağı kontrolü kullanılır. XML belgelerine veri erişimi sağlamak için **XMLDataSource** kontrolü kullanılır.

3.6.1. AccessDataSource

SqlDataSource kontrolünden türetilmiştir. **Microsoft.Jet.OLEDB.4.0** provider'ı kullanarak veri kaynağına bağlanır. Veri alışverişini gerçekleştirmek için SQL sorguları kullanır. **DataFile** özelliği kullanılarak veri alışverişinde bulunulacak veritabanı dosyası ve adresi belirtilir. Veri bağlanan kontrol AccessDataSource kontrolüne **DataSourceID** özelliği kullanılarak bağlanır.

AccessDataSource iki farklı veri kaynağı modunu destekler. Bunlar; **DataSet ve DataReader**'dir. DataSet kullanılarak hafızaya alınan veriler üzerinde sıralama (**sorting**) ve filtreleme (**filtering**) yapılabilir.

AccessDataSource ve GridView kontrollerini kullanarak örnek bir sayfa (**gridView.aspx**) oluşturmadan önce veritabanının izinlerini ayarlayalım.

Masaüstünden **Bilgisayarım** penceresini açıp, ardından **Araçlar** menüsünden **Klasör Seçenekleri...** komutunu veriniz. **Klasör Seçenekleri** penceresinde **Basit dosya paylaşımı kullan (önerilen)** ayarının önündeki onay işaretini kaldırıp **Tamam** düğmesini tıklatınız. Bu işlemden sonra **urunler.mdb** dosyasının kısayol menüsünden (sağ fare tuşunu kullanarak) **Özellikler** komutunu verip **Güvenlik** sekmesini tıklatınız. **Grup ya da kullanıcı adları** bölümünden **Users(Bilgisayar adı\Users)** seçeneğini seçip **Users için izinler** bölümünden **Tam Denetim (İzin Ver)** onay kutusunu seçiniz.

NOT: Güvenlik açısından her kullanıcıya veritabanlarına erişim izni verilmez. Duruma göre farklı kullanıcılara farklı yetkiler verilir.


```

9 <form runat="server" name="form1" method="post" action="">
10 <asp:AccessDataSource id="AccessDataSource1" runat="server"
11   DataSourceMode="DataSet"
12   DataFile="C:\inetpub\wwwroot\Site\data\urunler.mdb"
13   SelectCommand="SELECT Nu, Kitapadi, Yazaradi FROM Kitaplar"
14   UpdateCommand="UPDATE Kitaplar SET KitapAdi = @KitapAdi,
15   YazarAdi = @YazarAdi WHERE nu = @nu"
16   DeleteCommand="DELETE FROM Kitaplar WHERE nu = @nu"/>
17
18 <asp:GridView ID="GridView1" runat="server" DataSourceID="AccessDataSource1"
19   AutoGenerateColumns="False" AllowPaging="true" AllowSorting="true"
20   PageSize="3" DataKeyNames="Nu" AutoGenerateEditButton="true"
21   AutoGenerateDeleteButton="true">
22   <PagerSettings Mode="Numeric" />
23   <PagerStyle ForeColor="White" HorizontalAlign="Center" BackColor="#284775"/>
24   <Columns>
25     <asp:BoundField HeaderText="Sıra Nu" SortExpression="nu"
26       DataField="Nu" ReadOnly="true" />
27     <asp:BoundField HeaderText="Kitap Adı" SortExpression="Kitapadi"
28       DataField="Kitapadi" />
29     <asp:BoundField HeaderText="Yazar Adı" SortExpression="Yazaradi"
30       DataField="Yazaradi" />
31   </Columns>
32 </asp:GridView>
33 </form>

```

	Sıra Nu	Kitap Adı	Yazar Adı
Edit Delete	1	Nutuk	M.Kemal Atatürk
Edit Delete	2	Mesnevi'den Seçmeler	Mevlana
Edit Delete	3	Çalığışu	Reşat Nuri Güntekin

1 2 3

Resim 3.28: GridView.aspx sayfasının tarayıcıdaki görüntüsü

	Sıra Nu	Kitap Adı	Yazar Adı
Update Cancel	1	Nutuk	M.Kemal Atatürk
Edit Delete	2	Mesnevi'den Seçmeler	Mevlana
Edit Delete	3	Çalığışu	Reşat Nuri Güntekin

1 2 3

Resim 3.29: Düzenleme modunda GridView.aspx sayfasının tarayıcıdaki görüntüsü

GridView ile veri kaynağından gelen veriler tablo hâlinde görüntülenebilir, veriler üzerinde düzenleme ve sıralama işlemleri yapılabilir. ASP.NET 2.0 sürümüyle gelen bu kontrol DataGrid'in birçok özelliğini otomatik hâle getirmesiyle DataGrid'den daha gelişmiş bir kontroldür. Visual Studio 2005'te nerdeyse hiç kod yazmadan pencereler vasıtasıyla birçok özelliği kullanılabilir. Dreamweaver'da ise bu kontrole görsel destek olmadığından kod yazılarak kullanılabilir.

DataSourceMode="DataSet" ile **AccessDataSource**'un veri kaynağı modu DataSet olarak ayarlanmıştır. **DataFile** ile **urunler.mdb** dosyası veri dosyası olarak belirlenmiştir. **SelectCommand** ile **Kitaplar** tablosundan **Nu**, **KitapAdi**, **YazarAdi** alanları GridView'de görüntülenmek için seçilmiştir. **UpdateCommand** ile güncellenecek alanlar belirlenmiştir. **DeleteCommand** ile silme komutu verilmiştir.

DataSourceID ile GridView'in veri kaynağı kontrolü belirtilmiştir. **AutoGenerateColumns="False"** ile otomatik olarak sütunların oluşturulması pasif hâle getirilmiştir. Böylelikle sütunlar **Columns** koleksiyonu içerisinde **<asp:BoundField>** etiketleriyle oluşturulmuştur. **<asp:BoundField>** etiketi içinde **HeaderText** ile sütunların başlıklarında görünmesi istenilen metin, **DataField** ile sütunun hangi veri alanını görüntüleyeceği, **SortExpression** ile sıralama ifadesi, **ReadOnly** ile ise alanını yalnızca okunabilir veya güncellenebilir bir alan olduğu belirtilmiştir. **Columns** koleksiyonu içerisine **CheckBoxField**, **ImageField**, **HyperLinkField**, **CommandField**, **ButtonField**, **TemplateField** nesneleri eklenebilir.

AllowPaging="true" ile sayfalama, **AllowSorting="true"** ile sıralama işlemi etkin hâle getirilmiştir. Sıralamanın etkinleştirilmesi sütun başlıklarını link button (bağlantı düğmesi) haline getirmiştir. Sıralamanın yapılması için tıklandığında GridView ilgili **SortExpression**'ı veri kaynağı kontrolüne geçirir. Veri kaynağı kontrolü de sıralanmış veriyi GridView'e geri gönderir. GridView otomatik olarak artan ve azalan sıralama arasında geçiş yapar.

PageSize=3 ile sayfada gösterilecek kayıt sayısı 3 olarak ayarlanmıştır. **DataKeyNames** ile birincil anahtar (**primary key**) olan **nu** belirtilmiştir. **AutoGenerateEditButton** ile düzenleme amaçlı **Edit** yapısı GridView'e eklenmiştir. **Edit**'e tıklandığında **Update** ve **Cancel** bağlantı düğmeleri gelir, böylelikle düzenleme moduna (**Edit Mode**) geçilmiş olunur. Düzenleme modundayken **ReadOnly** özelliği **False** olan sütunlarda değişiklik yapılabilir.

AutoGenerateDeleteButton ile **Delete** link button GridView kontrolüne eklenmiştir.

<PagerSettings> etiketi içerisinde **Mode="Numeric"** ile sayfalara gitmek için sayfa numaraları eklenmiştir. **<PagerStyle>** etiketi içerisindeki **ForeColor**, **HorizontalAlign**, **BackColor** özellikleri ile biçim özellikleri belirlenmiştir.

```
SELECT Nu, Kitapadi, Yazaradi FROM Kitaplar
```

Edit Delete	2	Mesnevi'den Seçmeler	Mevlana
	Nu	Kitapadi	Yazaradi

Update Cancel	2	Mesnevi'den Seçmeler	Mevlana
	@nu	@Kitapadi	@Yazaradi

```
UPDATE Kitaplar SET Kitapadi = @Kitapadi, Yazaradi = @Yazaradi WHERE nu = @nu"
```

Resim 3:30: Alan isimleri ve parametreler arasındaki ilişki


Arama.aspx

```
7 <script runat="server">
8 sub Page_Load(sender as object, e as EventArgs)
9     If Not Page.IsPostBack Then
10         AccessDataSource1.SelectCommand = "SELECT * FROM kitaplar"
11         AccessDataSource1.SelectParameters("kitapadi").DefaultValue = ""
12     End If
13 End sub
14
15 Sub Button1_Click(sender as object, e as EventArgs)
16     Dim i as string
17     If (TextBox1.Text = "") Then
18         i = ""
19         AccessDataSource1.SelectCommand = "SELECT * FROM kitaplar"
20     Else
21         i = TextBox1.Text
22     End If
23     AccessDataSource1.SelectParameters("kitapadi").DefaultValue = i
24 End Sub
25 </script>
26 </head>
27 <body>
28 <form runat="server">
29 <asp:accessdatasource id="AccessDataSource1" runat="server"
30     datasourcemode="DataSet"
31     datafile="C:\Inetpub\wwwroot\Sitem\data\urunler.mdb"
32     selectcommand="SELECT * FROM Kitaplar WHERE Kitapadi
33         LIKE '%' + @kitapadi + '%"
34     updatecommand="Update kitaplar SET Nu=@Nu, Kitapadi=@Kitapadi,
35         Yazaradi=@Yazaradi WHERE nu=@nu">
36     <SelectParameters>
37         <asp:Parameter Name="Kitapadi" Type="String" />
38     </SelectParameters>
39 </asp:accessdatasource>
```

```

40
41 <asp:gridview id="GridView1"
42     runat="server"
43     autogeneratecolumns="False"
44     datakeynames="nu"
45     autogenerateeditbutton="True"
46     autogenerateddeletebutton="True"
47     datasourceid="AccessDataSource1">
48     <columns>
49         <asp:boundfield headertext="Sıra Numarası" datafield="nu"
50             InsertVisible="False" ReadOnly="True" SortExpression="Nu"/>
51         <asp:boundfield headertext="Kitap Adı" datafield="kitapadi"/>
52         <asp:boundfield headertext="Yazar Adı" datafield="yazaradi"/>
53     </columns>
54 </asp:gridview>
55 <asp:TextBox id="TextBox1" runat="server"/>
56 <asp:Button id="Button1" Text="Göster" onClick="Button1_Click"
57     runat="server"/>
58 </form>

```

Adres  http://localhost/sitem/arama.aspx

	Sıra Numarası	Kitap Adı	Yazar Adı
Edit Delete	1	Nutuk	M.Kemal Atatürk
Edit Delete	2	Mesnevi'den Seçmeler	Mevlana
Edit Delete	3	Çalığışu	Reşat Nuri Güntekin
Edit Delete	4	Dostlar Beni Hatırlasın	Aşık Veysel
Edit Delete	5	Yaşar Ne Yaşar Ne Yaşamaz	Aziz Nesin
Edit Delete	6	Memleketimden İnsan Manzaraları	Nazım Hikmet
Edit Delete	7	Türkçe'nin Sırları	Nihad Sami Banarlı
Edit Delete	8	Babalar ve Oğullar	Turgenyev

Baba

Resim 3.31: Arama.aspx sayfasının tarayıcıdaki görüntüsü

Adres  http://localhost/sitem/arama.aspx

	Sıra Numarası	Kitap Adı	Yazar Adı
Edit Delete	8	Babalar ve Oğullar	Turgenyev

Baba

Resim 3.32: Aranılan kaydı bulunması

anaAyrinti.aspx

```
7 <script runat="server">
8     Sub DVAYrintilar_ItemInserted(ByVal sender As Object, _
9         ByVal e As DetailsViewInsertedEventArgs)
10         ' DetailsView'de yeni kayıt eklendiğinde GridView'i yenileme
11         GVKitaplar.DataBind()
12     End Sub
13
14     Sub DVAYrintilar_ItemUpdated(ByVal sender As Object, _
15         ByVal e As DetailsViewUpdatedEventArgs)
16         ' DetailsView'de bir kayıt güncellendiğinde GridView'i yenileme
17         GVKitaplar.DataBind()
18     End Sub
19
20     Sub DVAYrintilar_ItemDeleted(ByVal sender As Object, _
21         ByVal e As DetailsViewDeletedEventArgs)
22         ' DetailsView'de bir kayıt silindiğinde GridView'i yenileme
23         GVKitaplar.DataBind()
24     End Sub
25 </script>
26 <html>
27 <body>
28     <form id="Form1" runat="server">
29         <h3>GridView ile DetailsView </h3>
30         <table cellpadding="10">
31             <tr>
32                 <td>
33                     <asp:GridView ID="GVKitaplar" DataSourceID="ADSKitaplar"
34                         AutoGenerateColumns="False" DataKeyNames="Nu" runat="server">
35                         <HeaderStyle BackColor="Blue" ForeColor="White" />
36                         <Columns>
37                             <asp:CommandField ShowSelectButton="True" />
38                             <asp:BoundField DataField="Nu" HeaderText="Sıra Nu" />
39                             <asp:BoundField DataField="Kitapadi" HeaderText="Kitap Adı" />
```

```

40         </Columns>
41     </asp:GridView>
42 </td>
43 <td valign="top">
44     <asp:DetailsView ID="DVAYrintilar" DataSourceID="ADSAYrintilar"
45     AutoGenerateRows="false"
46     AutoGenerateInsertButton="true"
47     AutoGenerateEditButton="true"
48     AutoGenerateDeleteButton="true"
49     EmptyDataText="Kayıt yok."
50     DataKeyNames="Nu" GridLines="Both"
51     OnItemInserted="DVAYrintilar_ItemInserted"
52     OnItemUpdated="DVAYrintilar_ItemUpdated"
53     OnItemDeleted="DVAYrintilar_ItemDeleted"
54     runat="server">
55     <HeaderStyle BackColor="Navy" ForeColor="White" />
56     <RowStyle BackColor="White" />
57     <AlternatingRowStyle BackColor="LightGray" />
58     <EditRowStyle BackColor="LightCyan" />
59     <Fields>
60         <asp:BoundField DataField="Nu" HeaderText="Nu"
61         ReadOnly="True" />
62         <asp:BoundField DataField="Kitapadi" HeaderText="Kitap Adı" />
63         <asp:BoundField DataField="Yazaradi" HeaderText="Yazar Adı" />
64         <asp:BoundField DataField="Yayinevi" HeaderText="Yayınevi" />
65     </Fields>
66     </asp:DetailsView>
67 </td>
68 </tr>
69 </table>
70
71 <asp:AccessDataSource ID="ADSKitaplar" runat="server"
72 DataSourceMode="DataSet"
73 DataFile="C:\Inetpub\wwwroot\Sitem\data\urunler.mdb"
74     SelectCommand="SELECT Nu, Kitapadi FROM Kitaplar">
75 </asp:AccessDataSource>
76
77 <asp:AccessDataSource ID="ADSAYrintilar" runat="server"
78 DataSourceMode="DataSet"
79 DataFile="C:\Inetpub\wwwroot\Sitem\data\urunler.mdb"
80     SelectCommand="SELECT * FROM Kitaplar WHERE Nu = @Nu"
81

```

```

82 DeleteCommand="DELETE FROM Kitaplar WHERE Nu = @Nu"
83
84 InsertCommand="INSERT INTO Kitaplar (Nu, Kitapadi, Yazaradi, Yayinevi)
85     VALUES (@Nu, @Kitapadi, @Yazaradi, @Yayinevi)"
86
87 UpdateCommand="UPDATE Kitaplar SET Kitapadi = @Kitapadi,
88     Yazaradi = @Yazaradi, Yayinevi = @Yayinevi WHERE Nu = @Nu" >
89
90 <SelectParameters>
91     <asp:ControlParameter ControlID="GVKitaplar"
92         Name="Nu" PropertyName="SelectedValue"
93         Type="String" />
94 </SelectParameters>
95
96 <DeleteParameters>
97     <asp:Parameter Name="Nu" Type="String" />
98 </DeleteParameters>
99
100 <UpdateParameters>
101     <asp:Parameter Name="Kitapadi" Type="String" />
102     <asp:Parameter Name="Yazaradi" Type="String" />
103     <asp:Parameter Name="Yayinevi" Type="String" />
104 </UpdateParameters>
105
106 <InsertParameters>
107     <asp:Parameter Name="Nu" Type="String" />
108     <asp:Parameter Name="Kitapadi" Type="String" />
109     <asp:Parameter Name="Yazaradi" Type="String" />
110     <asp:Parameter Name="Yayinevi" Type="String" />
111 </InsertParameters>
112
113 </asp:AccessDataSource>
114 </form>

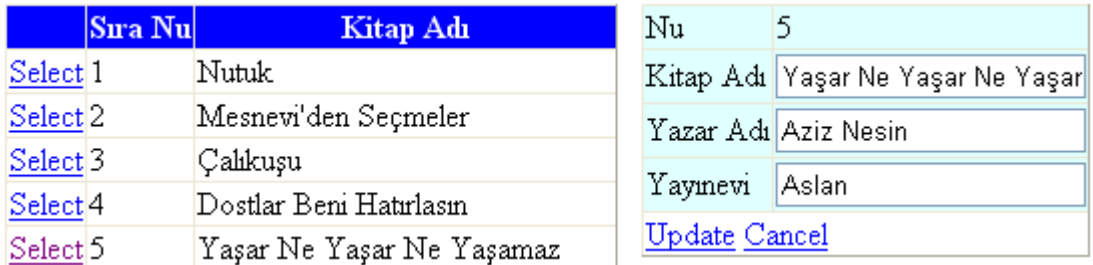
```



Resim 3.33: AnaAyrinti.aspx sayfasının tarayıcıdaki görüntüsü



Resim 3.34: "Select" tıklandığında



Resim 3.35: "Edit" tıklandığında



Resim 3.36: "New" tıklandığında

UYGULAMA FAALİYETİ

Ogrenciler veritabanında **siniflistesi** tablosundaki bilgilerin **GridView** kontrolü vasıtasıyla görüntülediği ve istenilen güncellemelerin yapıldığı ASP.NET sayfasını geliştiriniz.



The screenshot shows a web application window titled "siniflistesi : Tablo". It contains a GridView control displaying a table with 4 columns: "siranu", "ad", "soyad", and "cinsiyet". The table lists 16 students. Below the table, there is a search box labeled "(OtomatikSayı)" and a pagination control showing "Kayıt: 17 / 17".

siranu	ad	soyad	cinsiyet
1	Ezgi	Doğan	K
2	Coşkun	Çelik	E
3	Bahar	Akın	K
4	Esra	Yılmaz	K
5	Seda	Çoban	K
6	Cafer	Volkan	E
7	Ali	Rıza	E
8	Emre	Şener	E
9	Hasan	Çakar	E
10	Latife	Gündoğdu	K
11	Melis	Güner	K
12	Reyhan	Esmer	K
13	Feyza	Katar	K
14	Okan	Atar	E
15	Gülçin	Nazlı	K
16	Sinem	Sabah	K

İşlem Basamakları	Öneriler
Veri kaynağı kontrolünü oluşturunuz.	<asp:AccessDataSource...
Güncelleme işlemi yapabilmek için uygun veri kaynağı modunu seçiniz.	DataSourceMode...
Veri dosyasını seçiniz.	DataFile...
Seçme, güncelleme, silme SQL komutlarını oluşturunuz.	SelectCommand, UpdateCommand, DeleteCommand...
Veri bağlanılan kontrolü oluşturunuz.	GridView...
Veri bağlanılan kontrolü ile veri kaynağı kontrolü arasındaki ilişkiyi kurunuz.	DataSourceID...
İşlem Basamakları	Öneriler
Sütunları otomatik oluşturup oluşturmayacağınıza karar veriniz.	AutoGenerateColumns...
Sayfalama ve sıralama yapıp yapmayacağınıza karar veriniz.	AllowPaging, AllowSorting...
Birincil anahtarları belirtiniz.	DataKeyNames...
Düzenleme ve silme düğmelerinin otomatik oluşturulup oluşturulmayacağına karar veriniz.	AutoGenerateEditButton, AutoGenerateDeleteButton...
Sayfanın görünümüyle ilgili ayarları yapınız.	PagerSettings, PagerStyle...
Sütunları otomatik oluşturmadıysanız kodlarla oluşturunuz.	Columns... BoundField...

ÖLÇME VE DEĞERLENDİRME

Aşağıda verilen sorular için doğru cevap seçeneklerini işaretleyiniz.

- Veritabanına kayıt eklemek için kullanılan SQL deyimi nedir?
 - Select
 - Insert
 - Delete
 - Update
- Sadece veritabanındaki verileri okumak amacıyla kullanılan nesnenin adı nedir?
 - Connection
 - Command
 - DataReader
 - DataSet
- Aşağıdaki ifadelerden hangisi yanlıştır?
 - SQL ifadelerini veritabanına karşı yürütmek amacıyla Command nesnesi kullanılır.
 - DataSet'teki veri Fill metoduyla DataAdapter'e yüklenir.
 - DataSet nesnesi ile veri kaynağı arasında köprü vazifesi görür.
 - DataSet içindeki veriler web sunucusunun hafızasında tutulur.
- Aşağıdakilerden hangisi DataSet'in yapısı içinde bulunmaz?
 - DataTableConnection
 - DataTable
 - DataRowCollection
 - DataAdapter
- Bir kontrole veri bağlamak için kullanılan metot nedir?
 - DataRow
 - DataColumn
 - DataBind
 - DataConstraint

MODÜL DEĞERLENDİRME

OBJEKTİF TESTLER (ÖLÇME SORULARI)

1. Musteriler tablosundan Adı "Abidin" olan kişilerin kaydını getirin SQL deyimi aşağıdakilerden hangisidir?
A) SELECT * FROM Kisiler WHERE Ad="Abidin"
B) SELECT * FROM Kisiler
C) SELECT * FROM Kisiler WHERE "Abidin"
D) SELECT * FROM Kisiler Ad="Abidin"
2. Musteriler tablosunun Ad alanına Fatma ismini eklemek için kullanılan SQL deyimi aşağıdakilerden hangisidir?
A) INSERT Musteriler (Ad) VALUES ("Abidin")
B) INSERT INTO Musteriler VALUES ("Abidin")
C) INSERT INTO Musteriler (Ad) ("Abidin")
D) INSERT INTO Musteriler (Ad) VALUES ("Abidin")
3. Aşağıdakilerden hangisi DataSet'in yapısı içinde bulunmaz?
A) DataTable
B) DataRow
C) DataColumn
D) DataSource
4. Web sitesini yapılandırmak için kullanılan yapılandırma (konfigürasyon) dosyasının adı nedir?
A) Web.config
B) DreamweaverCtrls.dll
C) ADO.NET
D) System.data
5. Kontrole veri bağlanmasında kullanılan metot nedir?
A) DataRow
B) DataBind
C) DataColumn
D) DataConstraint

SÖZLÜK

Terim	Okunuşu	Anlamı, Açıklaması
Bind	Baynd	Bağlamak
Class	Klas	Sınıf
Click	Klik	Tıklamak
Configuration	Kınfigyireyşın	Yapılandırma, Konfigürasyon
Data	Deyta	Veriler, data. “Datum” kelimesinin çoğulu.
Datum	Deytım	Veri
Extension	Ekstenşın	Bir programın kullanımını iyileştiren ya da işlevini artıran program.
Event	İvent	Olay
Footer	futır	Sayfa altlığı. Bir yazılı belgede tek sayfalarda, çift sayfalarda ya da her sayfada yinelenen ve alt marjın üzerinde yer alan, belgenin adı ve sahibi, tarih gibi bilgileri belirten yazı bloğu.
Format	Formet	Biçim, format
Header	Hedır	Başlık.
Link	Link	Bağlantı
Layout	Leyaut	Yerleşim planı, serim. 1) Yazıcıda yazdırılacak verilerin sayfaya yerleştiriliş düzeni. 2) Birim işlemleri yapmak üzere tasarlanmış yapıtaş öğelerin düzenlenişi.
Page	Peyc	Sayfa
Random	Rendım	Rastgele, gelişigüzel, tesadüfi
Repeater	Ripitır	Tekrarlayıcı, yineleyici
Separator	Sepireytr	Ayıraç, ayırıcı. Karakter dizgilerinin sınırlarını belirten ayırıcı.
Set	Set	Set, küme
Style	Stayl	Stil, tarz, biçim
Template	Tempıleyt	Şablon
Validation	Velideyşın	Geçerlilik

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ – 1 CEVAP ANAHTARI

Sorular	Cevaplar
1	C
2	D
3	B
4	B
5	A

ÖĞRENME FAALİYETİ – 2 CEVAP ANAHTARI

Sorular	Cevaplar
1	D
2	C
3	B
4	B
5	A

ÖĞRENME FAALİYETİ – 3 CEVAP ANAHTARI

Sorular	Cevaplar
1	B
2	C
3	B
4	D
5	C

MODÜL DEĞERLENDİRME CEVAP ANAHTARI

Sorular	Cevaplar
1	A
2	D
3	D
4	A
5	B

KAYNAKÇA

- DEMİRKOL Zafer, ASP.NET, Pusula Yayıncılık, İstanbul, 2005.
- DUTHIE G. Andrew, Adım Adım Microsoft ASP.NET, Arkadaş Yayınevi, Ankara, 2005.
- SANKUR Bülent, Bilişim Sözlüğü 2005 Programı, Yazılım: Hakan GÜLERYÜZ, Pusula Yayıncılık, İstanbul, 2005.
- PALA Zeydin, ASP.NET İle Adım Adım Web Uygulamaları, Türkmen Kitabevi, İstanbul, 2006.
- Costas Hadjisotiriou, Kevin Marshall ve Rachel Andrew, ASP.NET Web Development with Macromedia Dreamweaver MX 2004, Apress.