

**T.C.
MİLLÎ EĞİTİM BAKANLIĞI**

BİLİŞİM TEKNOLOJİLERİ

ETKİLEŞİMLİ WEB UYGULAMALARINDA WEB FORMLARI

Ankara, 2013

- Bu modül, mesleki ve teknik eğitim okul/kurumlarında uygulanan Çerçeve Öğretim Programlarında yer alan yeterlikleri kazandırmaya yönelik olarak öğrencilere rehberlik etmek amacıyla hazırlanmış bireysel öğrenme materyalidir.
- Millî Eğitim Bakanlığınca ücretsiz olarak verilmiştir.
- **PARA İLE SATILMAZ.**

İÇİNDEKİLER

AÇIKLAMALAR	iii
GİRİŞ	1
ÖĞRENME FAALİYETİ-1	3
1. ANASAYFA KULLANMA	3
1.1. Anasayfa Oluşturma.....	4
1.2. Anasayfadan Web Sayfası Türetme	5
UYGULAMA FAALİYETİ	7
ÖLÇME VE DEĞERLENDİRME	9
ÖĞRENME FAALİYETİ-2	10
2. STİL SAYFALARI KULLANMA	10
2.1. Sayfa Öğelerini Ayrı Ayrı Biçimlendirme	10
2.2. Stil Dosyalarını Kullanma.....	11
2.2.1. Stil Sayfası Dosyası Oluşturma	11
2.2.2. Web Sitesinde Paylaşılmış Stiller Kullanma	12
UYGULAMA FAALİYETİ	14
ÖLÇME VE DEĞERLENDİRME	17
ÖĞRENME FAALİYETİ-3	18
3. TEMALARI KULLANMA	18
3.1. ASP.NET Temaları	18
3.1.1. Yeni Tema Oluşturma.....	18
3.1.2. Dış Görünüm Ekleme	19
3.1.3. CSS Ekleme	21
3.2. Temaların Uygulanması	21
3.2.1. Sayfaya Tema Uygulama.....	21
3.2.2. Siteye Tema Uygulama.....	22
UYGULAMA FAALİYETİ	24
ÖLÇME VE DEĞERLENDİRME	26
ÖĞRENME FAALİYETİ-4	27
4. SAYFA YÖNLENDİRME.....	27
4.1. Site Haritası.....	27
4.2. Menüler	30
4.2.1. Açılır ve Etkili Menüler	30
4.2.2. Ağaç Görünümlü Menüler	33
4.2.3. Site Haritası Yolu	36
UYGULAMA FAALİYETİ	37
ÖLÇME VE DEĞERLENDİRME	40
ÖĞRENME FAALİYETİ-5	41
5. DURUM YÖNETİCİSİ.....	41
5.1. Sorgulama Cümlesi (QueryString).....	41
5.2. Görünüm Durumu Yönetimi (ViewState).....	43
5.3. Çerezler (Cookies)	44
5.4. Oturum Yönetimi (Session Management)	45
5.5. Uygulama Durum Yönetimi (Application State Management)	48
UYGULAMA FAALİYETİ	50
ÖLÇME VE DEĞERLENDİRME	54
MODÜL DEĞERLENDİRME	55

CEVAP ANAHTARLARI.....	58
KAYNAKÇA.....	60

AÇIKLAMALAR

ALAN	Bilişim Teknolojileri
DAL/MESLEK	Web Programcılığı
MODÜLÜN ADI	Etkileşimli Web Uygulamalarında Web Formları
MODÜLÜN TANIMI	Bu modül, programlama yazılımı kullanarak .NET ortamında etkileşimli web uygulamalarında web formları ile ilgili temel bilgi ve becerilerin kazandırıldığı bir öğrenme materyalidir.
SÜRE	40/32
ÖNKOŞUL	Etkileşimli Web Uygulamaları İçin Temel İşlemler modülünü tamamlamış olmak
YETERLİK	Web sitesi yapısını oluşturmak
MODÜLÜN AMACI	Genel Amaç Bu modül ile gerekli ortam sağlandığında web form uygulamaları yapabileceksiniz. Amaçlar <ol style="list-style-type: none">1. Anasayfa oluşturabilecek ve diğer sayfalar ile bağlantı kurabileceksiniz.2. Stil sayfaları oluşturabilecek ve web sayfalarını bu dosya üzerinden biçimlendirebileceksiniz.3. Web site için tema oluşturabilecek ve siteye uygulayabileceksiniz.4. Site haritası ve web sayfalarına menü ekleyerek site kullanımını kolaylaştırabileceksiniz.5. Site ve web sayfaları üzerinde durum yönetimi sağlayabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	Ortam: Bilişim Teknolojileri laboratuvarı, işletme ortamı Donanım: Web programlama yazılımlarını çalıştırabilecek yeterlikte bilgisayar, İnternet bağlantısı
ÖLÇME VE DEĞERLENDİRME	Modül içinde yer alan her öğrenme faaliyetinden sonra verilen ölçme araçları ile kendinizi değerlendireceksiniz. Öğretmen modül sonunda ölçme aracı (çoktan seçmeli test, doğru-yanlış testi, boşluk doldurma, eşleştirme vb.) kullanarak modül uygulamaları ile kazandığınız bilgi ve becerileri ölçerek sizi değerlendirecektir.

GİRİŞ

Sevgili Öğrenci,

Bir web sitesinin işlevselliği, o siteyi ziyaret eden kullanıcı sayısı ile doğru orantılıdır. Web sitesi kullanıcılarının kaybedilmemesi veya artırılması için web sayfasının sürekli güncel tutulmasının yanında görünümünün de belirli aralıklarda değiştirilmesi faydası olacaktır.

Sayfa sayısı az olan sitelerin görünülerinin değiştirmek sorun teşkil etmese de gelişmiş web sitelerinin görünülerinin değiştirilmesi problem çıkarır.

Modül içinde ASP.NET web sitelerinin tasarımlarının ve görünülerinin değiştirilmesi için size kolaylık sağlayacak yapılara yer verilmiş, çeşitli örnek uygulamalarla konular desteklenmiştir.

ÖĞRENME FAALİYETİ-1

AMAÇ

Anasayfa oluşturabilecek ve diğer sayfalarla bağlantı kurabileceksiniz.

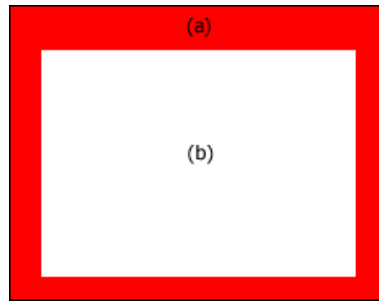
ARAŞTIRMA

- Bir web sayfasının her sayfasında değişmeden görüntülenen öğeleri araştırınız.

1. ANASAYFA KULLANMA

Bir web sitesi birden fazla sayfadan oluşur. Bu sayfalarda içerik değişirken her sayfada tekrarlanan öğeler ve özellikler vardır. Web sitesi hazırlanırken bir şablon oluşturularak şablon üzerinde içerik değişikliği ile sayfalar kolaylıkla hazırlanabilir. Ancak sayfa sayısı arttıkça web sitesinde güncelleme yapma bütün sayfaların tek tek elden geçirilerek değiştirilmesi anlamına gelmektedir. Bu durum zaman konusunda tasarımcılara çeşitli sıkıntılara sebep olacaktır.

CSS (*Cascading Style Sheets*) sayfaları kullanılarak web sitesinin sayfalarının özelliklerine doğrudan müdahale edilebilir. Bu şekilde sitenin tüm sayfalarının stil özellikleri tek bir yerden kontrol edilerek kolaylıkla güncelleme yapılabilir (CSS konusuna ilerleyen konularda yer verilecektir). Ancak tüm sayfalarda yer alan menü, resim, logo vb. içerik değişikliğinde bu yöntem işe yaramayacaktır. Daha önceki konularda bahsedilen Web Kullanıcı Kontrolleri ile ASP.NET'te kullanılarak bu duruma çözüm getirilmiştir. ASP.NET'in yeni sürümleri ile birlikte web kullanıcı kontrollerinin yanında *Anasayfa (MasterPage)* özelliği eklenerek tasarımcılara büyük kolaylıklar sağlanmıştır.



Resim 1.1: MasterPage Sayfası (a) Sabit Alanlar (b) İçerik Alanı

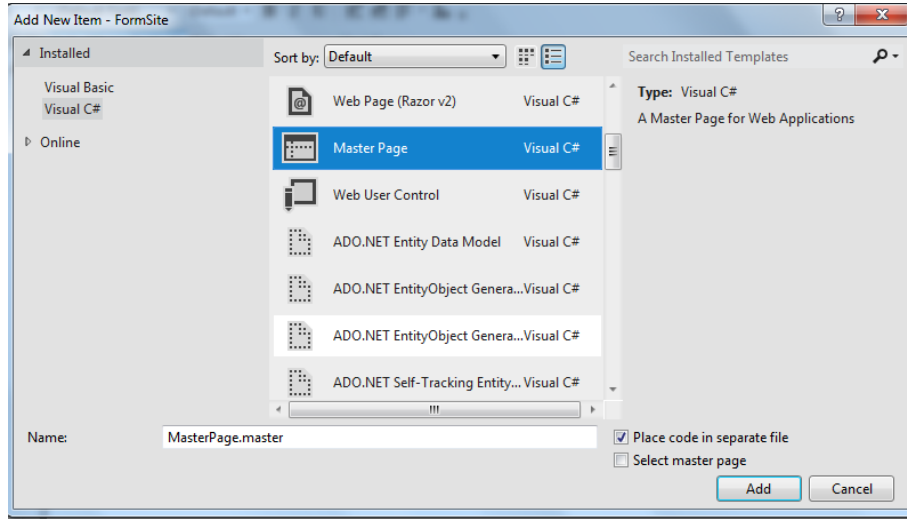
MasterPage kullanımı ile birlikte sitenin tasarımı oluşturulmaktadır. *MasterPage* üzerinde sabit kalması ve değişmesi istenen alanlar belirlenir. *MasterPage* kullanılarak siteye yeni eklenen bir sayfa tasarım uygulanır. *MasterPage* üzerinde yapılan bir değişiklik programlama yazılımı ile *MasterPage* kullanılan tüm sayfalarda otomatik olarak güncellenmektedir.

1.1. Anasayfa Oluřturma

MasterPage sayfaları standart ASP.NET sayfaları ile benzerlik göstermektedir. Dosya uzantısı *.master*'dir. ASP.NET sayfaları *@Page* direktifi ile başlarken *MasterPage* sayfaları *@Master* direktifi ile başlamaktadır. Temel bu iki fark haricinde tasarımı standart ASP.NET sayfaları ile aynıdır.

Web sitesine bir *MasterPage* eklemek için;

- *Add New Item – FormSite* iletişim penceresinden *MasterPage* komutunu seçin.



Resim 1.2: MasterPage sayfasının eklenmesi

MasterPage sayfasında tüm sayfalarda görüntülenmesi istenen alanlar belirlenebilir. Sayfalara özgü alanlar ise *ContentPlaceHolder* kontrolü ile belirlenir. *ContentPlaceHolder* kontrolü, *MasterPage* sayfalarından türeyen sayfalarda içeriği deęişen alanları ifade eder. Programlama yazılımı yeni eklenen *MasterPage* sayfasına iki tane *ContentPlaceHolder* kontrolü ekler. Birincisi sayfa tanımlamaların yapılması için *Head* etiketi içinde, ikincisi ise içeriğin eklenmesi için *Body* etiketi içindedir.

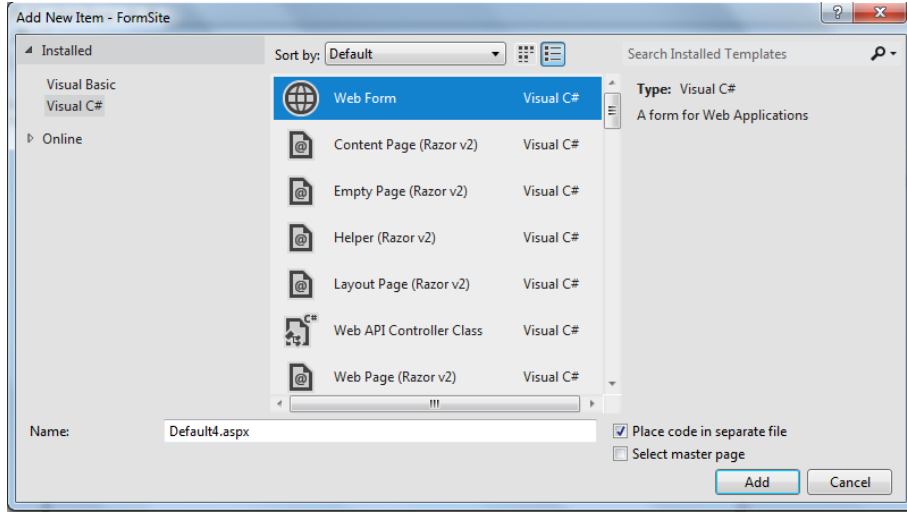
```
<head runat="server">
  <title></title>
  <asp:ContentPlaceHolder id="head" runat="server">
</asp:ContentPlaceHolder>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">
</asp:ContentPlaceHolder>
    </div>
  </form>
</body>
```

1.2. Anasayfadan Web Sayfası Türetme

Bir web projesinde birden fazla *MasterPage* kullanılabilir. Programlama yazılımı web sitesine yeni bir web form eklerken *MasterPage* sayfalarından birinin seçilerek sayfanın türetilmesine olanak sağlar.

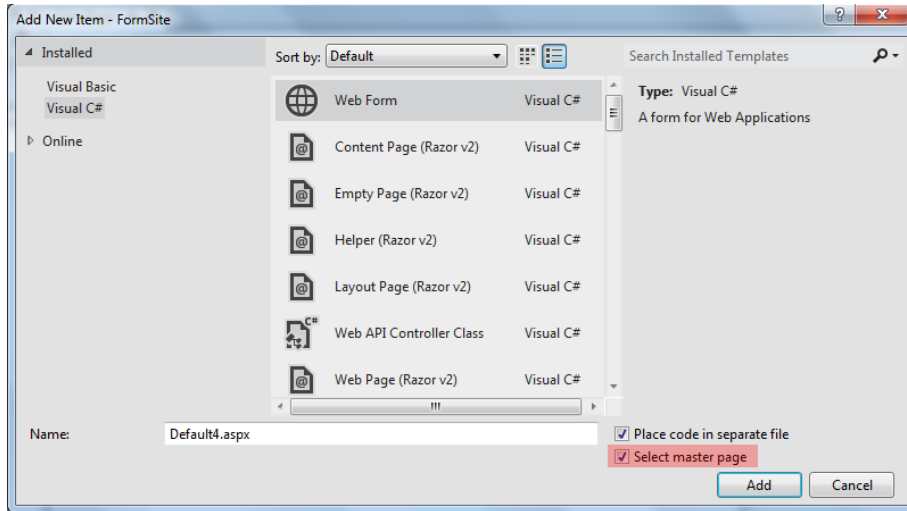
MasterPage sayfasından sayfa türetmek için;

- *Add New Item – FormSite* iletişim penceresinden *Web Form* komutunu seçin.



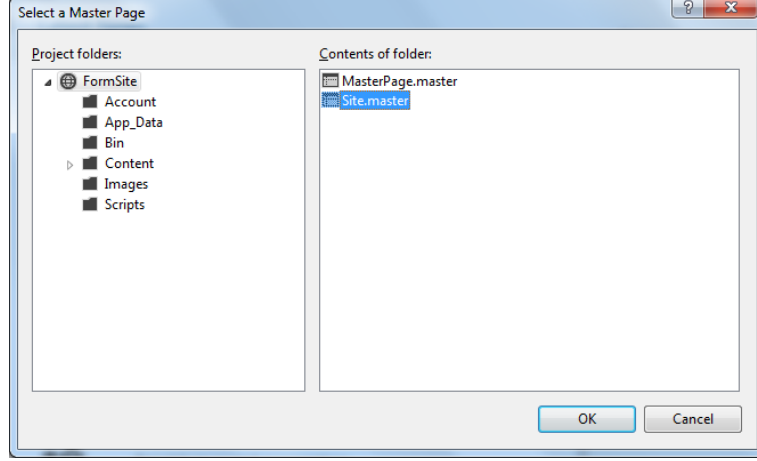
Resim 1.3: Web form eklenmesi

- Web form komutu seçildikten sonra *Add New Item – FormSite* iletişim penceresi sol alt bölümünde *Select Master Page* seçeneği aktif hale gelecektir. Bu alanı işaretleyin.



Resim 1.4: Select MasterPage seçeneği

- *Add* butonuna tıklayın. Ekrana gelen *Select a Master Page* iletişim penceresinden uygun *MasterPage* sayfasını seçin ve onaylayın.



Resim 1.5: Select a Master Page iletişim penceresi

MasterPage sayfasından sayfa üretmenin diğer bir yolu ise, *MasterPage* sayfası üzerinde sağ tıklayıp *Add Content Page* komutunun seçilmesidir. Bu durumda programlama yazılımı otomatik olarak seçilen *MasterPage* sayfasından türemiş yeni bir sayfa web sitesine ekleyecektir.

MasterPage sayfasından türemiş olan sayfanın kodları aşağıdaki gibi olacaktır. Kodlar incelenecek olursa `@Page` direktifi ile başlayan satırda hangi *MasterPage* sayfasından türetildiği belirtilmektedir. Ayrıca *MasterPage* sayfasında *Form* etiketi kullanıldığı için türeyen sayfa kodlarında *Form* etiketi yer almaz.

```
<%@ Page Title="" Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true" CodeFile="Default4.aspx.cs" Inherits="Default4" %>
```

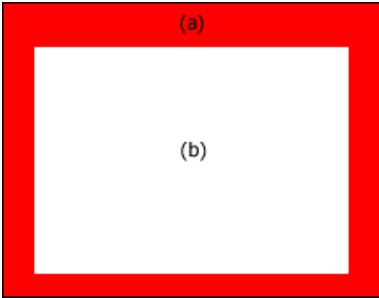
```
<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
```

```
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server"></asp:Content>
```

MasterPage sayfaları tarayıcıda tek başlarına görüntülenmez. *MasterPage* görüntülenmek istenmesi durumunda *MasterPage* sayfasından türeyen bir sayfa ile görüntülenmelidir.

UYGULAMA FAALİYETİ

Anasayfa kullanımı ile ilgili aşağıdaki uygulamayı yapınız.

İşlem Basamakları	Öneriler
<p>➤ Yeni boş bir web sitesi oluşturun.</p>	<p>➤ <i>File > New > Web Site</i> komutu seçebilirsiniz.</p> <p>➤ <i>New Web Site</i> iletişim penceresinden <i>ASP.NET Empty Web Site</i> komutunu seçebilirsiniz.</p>
<p>➤ Siteye yeni bir Anasayfa ekleyin.</p>	<p>➤ <i>Solution Explorer</i> panelinden site adını sağ tıklayarak <i>Add > Add New Item</i> ya da <i>Web Site > Add New Item</i> komutunu kullanabilirsiniz.</p> <p>➤ <i>Add New Item</i> iletişim penceresinden <i>Master Page</i> komutunu kullanabilirsiniz.</p>
<p>➤ Anasayfa görünümünü aşağıdaki gibi değiştirin.</p> 	<p>➤ Tasarım için tablodan faydalanabilirsiniz.</p> <p>➤ (a) sabit alanları, (b) sayfalarda değişecek alanları ifade etmektedir.</p>
<p>➤ Sabit alanları birbirinden ayırt etmek için isimlendirin.</p>	<p>➤ Üst sabit alan için <i>ÜST ALAN</i>, sağ alan için <i>SAG ALAN</i>, sol alan için <i>SOL ALAN</i> ve alt alan için <i>ALT ALAN</i> isimlerini kullanabilirsiniz.</p>
<p>➤ Anasayfayı kaydedin ve bu sayfadan iki tane sayfa türetin.</p>	<p>➤ <i>Add New Item</i> iletişim penceresinden <i>Web Form</i> komutunu seçebilirsiniz.</p> <p>➤ <i>Select Master Page</i> seçeneğini seçebilir ve sonraki adımda anasayfayı seçebilirsiniz.</p> <p>ya da</p> <p>Master Page seçili iken <i>Web Site > Add Content Page</i> komutunu seçebilirsiniz.</p>
<p>➤ Sayfaları çalıştırın ve görünümü inceleyin.</p>	<p>➤ <i>Debug > Start Debugging (F5)</i> komutunu kullanabilirsiniz.</p>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri		Evet	Hayır
1.	Anasayfa oluşturabildiniz mi?		
2.	Anasayfa üzerinde istediğiniz tasarımı gerçekleştirebildiniz mi?		
3.	Anasayfadan sayfa türetebildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda “**Hayır**” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “**Evet**” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. () Anasayfa programlama yazılımında *MasterPage* olarak adlandırılmaktadır.
2. () Bir web sitesine bir tane *MasterPage* eklenebilir.
3. () Bir *MasterPage* üzerinde sayfalarda değişen ve sabit kalan alanlar belirlenebilir.
4. () *MasterPage* üzerinde yapılan değişiklik diğer sayfalara tek tek yansıtılmalıdır.
5. () *Masterpage* kodları *@Page* direktifi ile başlar.
6. () Sayfalarda değişen alanlar *ContentPlaceHolder* kontrolü ile belirlenir.
7. () Bir *MasterPage*'de birden fazla *ContentPlaceHolder* kontrolü eklenebilir.
8. () Türetilen sayfanın kodlarında hangi *MasterPage* sayfasından türetildiği *@Page* direktifi içinde belirtilir.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-2

AMAÇ

Stil sayfaları oluşturabilecek ve diğer sayfalar ile bağlantı kurabilecektir.

ARAŞTIRMA

- Stil sayfalarının HTML web sitelerinde kullanımını araştırınız.

2. STİL SAYFALARI KULLANMA

Stil sayfaları kullanılarak bir web sitesindeki HTML kontrolleri ve HTML etiketlerinin sayfaların kullanıcılara nasıl görüneceği ayrı bir dosyada tutulabilir. CSS (*Cascading Style Sheets*) olarak adlandırılan bu dosyaların güncellenmesi ile sitede bu stili kullanan tüm sayfalar otomatik olarak güncellenir.

2.1. Sayfa Öğelerini Ayrı Ayrı Biçimlendirme

Web sayfalarında kullanılan öğeler diğer öğelerden bağımsız olarak biçimlendirilebilir. Öğeleri biçimlendirmek için *Properties* paneli veya sayfa kodları kullanılabilir.

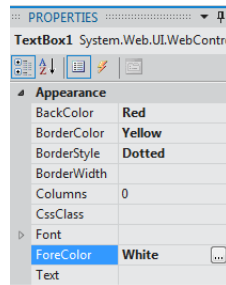
Sayfa öğelerini ayrı ayrı biçimlendirmek için;

- Yeni bir sayfa açın ve sayfaya aşağıdaki sunucu kontrollerini ekleyin.



Resim 2.1: Stil uygulanacak sunucu kontrolleri

- Birinci kontrolü seçin ve *Properties* panelinden özelliklerini değiştirin.

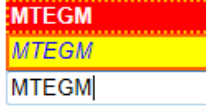


Resim 2.2: TextBox1 özellikleri

- İkinci kontrol özelliklerini kodlarla değiştirin.

```
<asp:TextBox ID="TextBox2" runat="server" BackColor="Yellow"
BorderColor="Red" BorderStyle="Double" ForeColor="Blue" Font-
Italic="True"></asp:TextBox>
```

- Üçüncü kontrolde değişiklik yapmayın.
- Uygulamayı çalıştırın ve kontrol görünümünü inceleyin.



Resim 2.3: Biçimlendirme uygulanmış sunucu kontrolleri

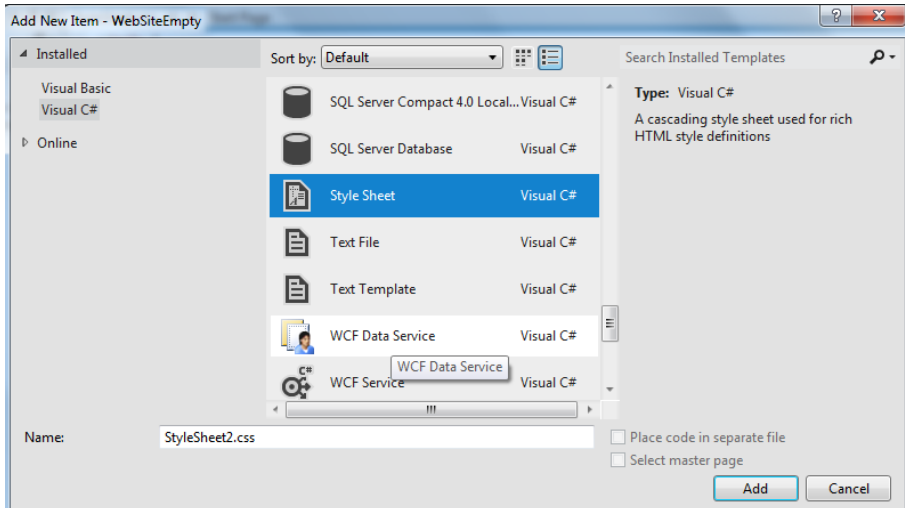
2.2. Stil Dosyalarını Kullanma

Bir önceki konuda yer alan öğelerin her birinin ayrı ayrı biçimlendirilmesi oldukça zahmetli ve zaman alan bir yöntem olacaktır. Programlama yazılımı geliştiricilere stil dosyalarını ASP.NET'te kullanarak daha kolay tasarım yapmalarını sağlamaktadır.

2.2.1. Stil Sayfası Dosyası Oluşturma

Web sitesine yeni bir stil dosyası eklemek için;

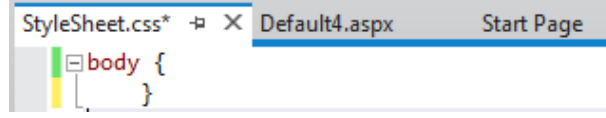
- *Add New Item* iletişim penceresinden *Style Sheet* komutunu seçin.



Resim 2.4: Stil dosyası eklenmesi

- Stil dosyasına bir ad verin ve onaylayın.

- *Solution Explorer* panelinden stil dosyasını isterseniz bir klasör altına taşıyabilirsiniz.
- Stil sayfası ekrana gelecektir. Stil sayfası ilk eklendiğinde stil sayfasında sadece *body* etiketi ile açılır.



Resim 2.5: Stil sayfası dosyası

- Yeni stil kuralına bir isim verin. Stil isminin başında nokta kullanmayı unutmayın.

```
.textbox {
}
```

- Programlama yazılımı kod tamamlama (intellisense) özelliği ile rahatlıkla stil sayfasında kural oluşturulabilir.



Resim 2.6: Kod tamamlama (Intellisesnse) özelliği

- Stil parantezleri arasına aşağıdaki kuralları yazın.

```
.textbox {
    background-color: Red;
    border-color: #f8f305;
    border-style: double;
    font-family: Arial;
    font-size: medium;
}
```

- Stil dosyasını kaydedin.

2.2.2. Web Sitesinde Paylaşılmış Stiller Kullanma

Hazırlanan bir stili sayfalarda kullanmak için;

- Stil ekleyeceğiniz sayfayı açın.
- Sayfaya stil uygulamak için iki tane *TextBox* kontrolü ekleyin.

- `<head>...</head>` etiketleri arasında stil dosyasını tanımlamak için aşağıdaki kodları ekleyin.

```
<link rel="stylesheet" type="text/css" href="StyleSheet.css" />
```

- Kontrollerin her biri için eğer bir CSS sınıf kuralı kullanacak ise mutlaka `CssClass` özelliği ile tanımlanmalıdır.

```
<asp:TextBox ID="TextBox2" runat="server" CssClass="textbox">  
</asp:TextBox>
```

- Stil uygulanan kontrol aşağıdaki gibi görünecektir.

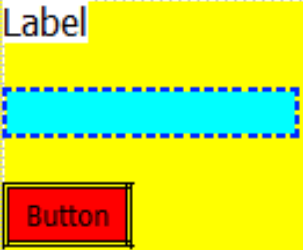


Resim 2.7: Stil uygulanmış kontrolün görünümü

UYGULAMA FAALİYETİ

Bir stil dosyası oluşturarak web sayfasına bağlayınız.

İşlem Basamakları	Öneriler
<p>➤ Yeni bir boş web projesi oluşturun.</p>	<p>➤ <i>File > New > Web Site</i> komutunu kullanabilirsiniz.</p> <p>➤ <i>New WebSite</i> iletişim penceresinden <i>ASP.NET Empty Web Site</i> komutunu kullanabilirsiniz.</p>
<p>➤ Siteye yeni bir stil sayfası ekleyin.</p>	<p>➤ <i>Solution Explorer</i> panelinden site adını sağ tıklayarak <i>Add > Add New Item</i> ya da <i>Web Site > Add New Item</i> komutunu kullanabilirsiniz.</p> <p>➤ <i>Add New Item</i> iletişim penceresinden <i>Style Sheet</i> komutunu kullanabilirsiniz.</p>
<p>➤ Sayfa stili kurallarını ekleyin.</p> <pre>body { background: yellow; font-family: Tahoma; }</pre>	<p>➤ Kod tamamlama özelliği ile kolaylıkla kod ekleyebilirsiniz.</p>
<p>➤ Button kontrolü için kuralları belirleyin.</p> <pre>.button { background-color: Red; border-color: #000; border-style: double; font-family: Tahoma; }</pre>	<p>➤ Kod tamamlama özelliği ile kolaylıkla kod ekleyebilirsiniz.</p>
<p>➤ Textbox kontrolü için kuralları belirleyin.</p> <pre>.textbox { background-color:#0ff; border-color:#0026ff; border-style: dashed; font-family: Tahoma; }</pre>	<p>➤ Kod tamamlama özelliği ile kolaylıkla kod ekleyebilirsiniz.</p>
<p>➤ Label kontrolü için kuralları belirleyin.</p> <pre>.label { background-color:#fff; border-color:red; font-family: Tahoma; }</pre>	<p>➤ Kod tamamlama özelliği ile kolaylıkla kod ekleyebilirsiniz.</p>

<p>➤ Sayfaya kontrolleri ekleyin.</p> <p>➤ Sayfaya stil dosyasını bağlayın.</p> <pre><link rel="stylesheet" type="text/css" href="StyleSheet.css" /></pre>	<p>➤ <i>Toolbox</i> panelini kullanabilirsiniz.</p> <p>➤ Sayfa kodlarında <code><head>...</head></code> etiketleri arasında dosyayı tanımlayabilirsiniz.</p>
<p>➤ Kontrollere stilleri atayın.</p>	<p>➤ <i>Properties</i> panelinde <i>CssClass</i> özelliğini ya da kontrol kodlarına <i>CssClass</i> etiketini kullanabilirsiniz.</p>
<p>➤ Uygulamayı çalıştırın ve kontrollerin stillerini gözlemleyin.</p> 	<p>➤ <i>Debug</i> > <i>Start Debugging</i> (F5) komutunu kullanabilirsiniz.</p>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Yeni web projesi oluşturabildiniz mi?		
2. Stil sayfası ekleyebildiniz mi?		
3. Stil kurallarını ekleyebildiniz mi?		
4. Sayfaya stil dosyasını bağlayabildiniz mi?		
5. Kontrollere stil atayabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda “**Hayır**” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “**Evet**” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. () Stil dosyası değiştirildiğinde bu stili kullanan tüm sayfalar otomatik olarak güncellenir.
2. () Stil dosyası kullanmadan sayfada kullanılan öğeler *Properties* paneli kullanılarak ayrı ayrı biçimlendirilebilir.
3. () Stil dosyaları hazırlanırken kod tamamlama özelliği kullanılmaz.
4. () Stil dosyasına yeni kural eklenirken kural isminin önüne . (nokta) işareti konur.
5. () Stil dosyaları sayfaya bağlanırken `<body>...</body>` etiketleri arasında tanımlanır.
6. () Kontrollere stil tanımlanırken *CSSClass* özelliği kullanılır.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-3

AMAÇ

Web sayfası için tema oluşturabilecek ve siteye uygulayabileceksiniz.

ARAŞTIRMA

- Web sitelerinde görünümün ara ara değiştirilmesinin faydalarını araştırınız.

3. TEMALARI KULLANMA

3.1. ASP.NET Temaları

Bir web sayfanın işlevselliği siteyi ziyaret eden kullanıcı sayısı ile doğru orantılıdır. Bu nedenle sitenin görünümü büyük önem taşımaktadır. Sitenin görünümünün iyi olmasının yanında görünümün ara ara değiştirilmesi siteyi ziyaret eden kullanıcı sayısını olumlu yönde etkileyecektir.

Sayfa sayısı az olan bir sitenin görünümünü değiştirmek pek sorun teşkil etmeyebilir, ancak sayfa sayısı arttıkça bir sitede görünümü değiştirmek oldukça büyük problemlere yol açacaktır. Bu durum *CSS (Cascading Style Sheet)* ile giderilse de ASP.NET'te yeterli olmamaktadır.

ASP.NET'te genel olarak bir sitenin görünümünün nasıl olacağını belirleyen yapılar yani temalar geliştirilmiştir. Temalar web sitesindeki tüm sayfalara *web.config* dosyasına yazılacak kodlarla uygulanabilmektedir.

3.1.1. Yeni Tema Oluşturma

App_Themes klasörü ASP.NET'te temaları saklamak için kullanılır. *App_Themes* klasörü altında yer alan klasörler de ASP.NET özel klasörleridir ve *Theme* klasörü olarak adlandırılır. Uygulamaya *Theme* klasörleri eklendikten sonra klasör içine *SKIN* dosyaları, *CSS* dosyaları ve resimler eklenebilir.

Yeni bir tema oluşturmak için;

- Solution Explorer panelinde, web sitesinin adını sağ tıklayın ve *Add > ASP.NET Folder > Theme* komutunu seçin.
- *App_Themes* klasörü eklenmiş durumda değilse programlama yazılımı otomatik olarak ekleyecektir.

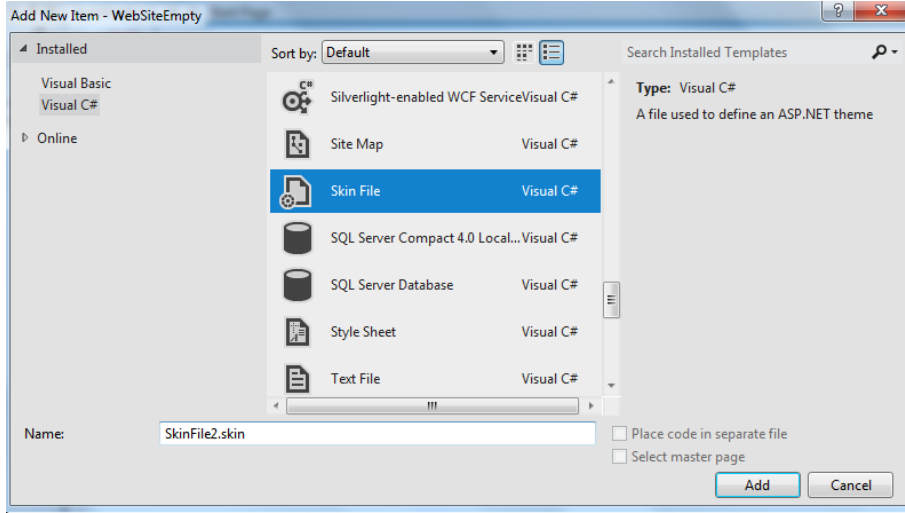
- Tema klasörüne dış görünüm dosyası, stil dosyaları (CSS) ve resimler ekleyebilirsiniz.

3.1.2. Dış Görünüm Ekleme

Theme klasörüne *skin* dosyaları eklenerek kontrollerin dış görünüşleri değiştirilebilir. *Skin* dosyaları ASP.NET sunucu kontrollerinin stillerini barındıran dosyalardır. Bir tema tanımlandığında sayfaya istek geldiğinde sayfalar kullanıcıya gönderilirken kontroller *skin* dosyalarında tanımlı stiller uygulanarak gönderilir. Bir *skin* dosyasında bir kontrol için birden fazla stil uygulanabilir.

Temaya bir *skin* dosyası eklemek için;

- *Add New Item* iletişim penceresinden *Skin File* komutunu seçin.



Resim 3.1: Skin dosyası ekleme

- *Skin* dosyasına bir ad verin ve onaylayın.
- *Skin* dosyası eklendikten sonra örnek olarak iki skin eklenecektir. Bunlar *GridView* ve *Image* kontrolü için tanımlanmıştır.

<%--

Default skin template. The following skins are provided as examples only.

1. Named control skin. The *SkinId* should be uniquely defined because duplicate *SkinId*'s per control type are not allowed in the same theme.

```
<asp:GridView runat="server" SkinId="gridviewSkin" BackColor="White" >
  <AlternatingRowStyle BackColor="Blue" />
</asp:GridView>
```

2. Default skin. The SkinId is not defined. Only one default control skin per control type is allowed in the same theme.

```
<asp:Image runat="server" ImageUrl="~/images/image1.jpg" />
--%>
```

- Kodlar incelenecek olursa *skin* dosyalarındaki kullanılan kodların kontrol kodları ile benzerlik gösterdiği görülecektir. Farklı olarak *ID* özelliği yerine *SkinID* özelliği yer almaktadır.
- *SkinID* özelliği kullanımı zorunlu değildir. *SkinID* özelliği tanımlanmamışsa *Default Skin*, tanımlanmışsa *Named Skin* olarak adlandırılır. Bu özelliğin kullanımı ilerleyen konularda anlatılacaktır.
- Örnek bir butona ait *skin* kodlaması aşağıdaki gibidir.

```
<asp:Button runat="server"
  BackColor="Red"
  ForeColor="Yellow"
  Font-Name="Arial"
  Font-Size="14px" />
```

- Programlama yazılımı ortamında *skin* dosyalarının kodlaması sırasında kod tamamlama (intellisense) özelliği çalışmaz. *Skin* dosyalarında daha hızlı ve doğru kodlama yapılabilmesi için sayfa kodlarından bir kontrole ait özellikler belirlendikten sonra bu kodlar *skin* dosyasına kopyalanıp *ID* özelliğinin silinmesi yeterli olacaktır.

```
<asp:TextBox runat="server"
  BackColor="Red"
  BorderColor="Yellow"
  BorderWidth="1px"
  Font-Bold="True"
  Font-Italic="True"
  ForeColor="White"></asp:TextBox>
```

Bir uygulamaya birden fazla *skin* dosyası eklenebilir. Burada dikkat edilmesi gereken *skin* dosyalarının adlandırılmasında aynı ismin kullanılmamasıdır. Kod karmaşıklığının önüne geçilmesi için her kontrol için ayrı *skin* dosyası oluşturulabilir.

3.1.3. CSS Ekleme

Skin dosyaları sadece kontrollerin dış görünümünün ayarlanması için kullanılmaktadır. Ancak temalarla birlikte bütün sayfada değişiklik yapılması istenebilir. Bu durumda HTML kontrolleri ve HTML etiketlerinin görünümünü ayarlamak için CSS dosyaları kullanılır.

Temaya bir CSS dosyası eklemek için;

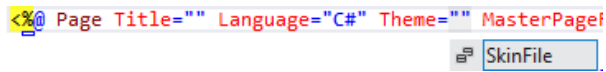
- *Add New Item* iletişim penceresinden *Style Sheet* komutunu seçin.
- CSS dosyasına bir ad verin ve onaylayın.
- CSS dosyasını *Solution Explorer* panelinden tema klasörünün altına taşıyın.
- CSS dosyasını önceki konularda anlatıldığı şekilde kodlayabilirsiniz.

3.2. Temaların Uygulanması

Oluşturulan temalar sitenin tamamına ya da sayfa düzeyinde uygulanabilir. Daha önceki uygulamalarda hatırlanacağı gibi sitenin tamamı için *web.config* dosyası, sayfa bazında ise *@Page* direktifi kullanılır.

3.2.1. Sayfaya Tema Uygulama

Sayfa düzeyinde tema uygulamak için *@Page* direktifi altındaki *Theme* özelliği kullanılır. *Theme* özelliği eklendikten sonra programlama yazılımı uygulamadaki temaları otomatik olarak listeleyecektir.



```
<%@ Page Title="" Language="C#" Theme="" MasterPageF  
SkinFile
```

Resim 3.2: Sayfa düzeyinde tema uygulama

Sayfaya tema uygulandıktan sonra, *skin* dosyasında tanımlanan özellikler kontrollere, CSS ile tanımlanan özellikler ise HTML kontrollerine ve HTML etiketlerine otomatik olarak uygulanacaktır.

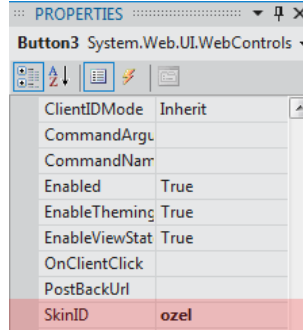
SkinID özelliği ile bir kontrole *Named Skin* uygulanabilir. Sayfaya tema uygulanmış olsa bile *Named Skin* çalışacaktır.

Bir kontrole *Named Skin* uygulamak için;

- Kontrol özellikleri kodlayın ve *SkinID* atayın.

```
<asp:Button SkinID="ozel" runat="server"
    BackColor="Yellow"
    ForeColor="Red"
    Font-Name="Arial"
    Font-Size="14px" />
```

- Properties paneli *Skin ID* özelliğine *Skin* dosyasında tanımladığımız ismi girin.



Resim 3.3: SkinID özelliğinin ayarlanması

Bazı durumlarda sayfaya tema uygulanmasına rağmen bir kontrole temanın uygulanması istenmeyebilir. Bir kontroldeki tema uygulamasını iptal etmek için *EnableTheming* özelliğinin *False* olarak ayarlanması yeterli olacaktır. Bu durumda kontrol varsayılan görünümü ile görüntülenir.

Sayfanın tasarım aşaması sırasında tema belirlenebileceği gibi çalışma zamanında da tema ataması yapılabilir. Kullanıcılar çalışma zamanında siteyi bu özellik sayesinde kendi isteklerine göre kişiselleştirebilecektir. *SkinID* özelliği ile bir kontrole ait birden fazla tema belirlenebileceği unutulmamalıdır.

Sayfanın temasının çalışma zamanında belirlenebilmesi için kontroller oluşturulmadan gerekli kodlar *Pre_Init* olayına yazılmalıdır.

```
void Page_PreInit(object sender, EventArgs e)
{
    Page.Theme = "SkinFile2";
}
```

3.2.2. Siteye Tema Uygulama

Site düzeyinde tema belirlemek için *Web.config* dosyasına ekleme yapılmalıdır `<System.web>...</System.web>` etiketleri arasına `<pages>` etiketi ile tema eklenebilir. Böylece sayfalara tek tek tema uygulamak yerine sitenin tamamına tema uygulanmış olmaktadır.

```
<system.web>  
...  
<pages theme="tema1"/>  
...  
</system.web>
```

Yukarıdaki kodlar *Web.config* dosyasına eklendikten sonra sitenin teması *Tema1* olarak otomatik olarak ayarlanır. Eğer site teması haricinde bir sayfaya aynı zamanda sayfa düzeyinde de tema uygulanmışsa sayfa düzeyindeki tema geçerli olacaktır.

Siteye tema uygulandıktan sonra bazı sayfalarda temanın uygulanması istenmeyebilir. Bu durumda *@Page* direktifi *EnableTheming* özelliği *False* olarak ayarlanmalıdır.

UYGULAMA FAALİYETİ

ASP.NET web sitesinde tema kullanımı ile ilgili aşağıdaki uygulamayı yapınız.

İşlem Basamakları	Öneriler
<p>➤ Uygulama faaliyeti 1'deki web sitesini açın.</p>	<p>➤ <i>Start Page</i> penceresini kullanabilirsiniz.</p>
<p>➤ Uygulamaya bir tema klasörü ekleyin.</p>	<p>➤ <i>Solution Explorer</i> panelinde site adını sağ tıklayarak <i>Add > ASP.NET Folder > Theme</i> komutunu seçebilirsiniz.</p>
<p>➤ Temaya bir dış görünüm dosyası ekleyin.</p>	<p>➤ <i>Add New Item</i> iletişim penceresinden <i>Skin File</i> komutunu seçebilirsiniz.</p>
<p>➤ Tema dış görünüm dosyasına aşağıdaki kodları yazın.</p> <pre><asp:CheckBox runat="server" BackColor="Red" BorderColor="Yellow" BorderStyle="Dotted" Font-Bold="True" Font-Size="Smaller" Height="20px" Width="125px" /></pre>	<p>➤ Kontrolü sayfa ekledikten sonra özellikleri <i>Properties</i> panelinden ayarlayabilirsiniz.</p> <p>➤ Özellikler <i>skin</i> dosyasına <i>id</i> özelliğini silebilirsiniz.</p>
<p>➤ Yeni bir sayfa oluşturun ve temayı sayfaya uygulayın.</p> <pre><%@ Page Language="C#" AutoEventWireup="true" Theme="Theme1" CodeFile="Default.aspx.cs" Inherits="Default" %></pre>	<p>➤ <i>@Page</i> direktifi <i>Theme</i> özelliğini kullanabilirsiniz.</p>
<p>➤ Sayfaya bir <i>CheckBox</i> kontrolü ekleyin.</p> <p><input type="checkbox"/> [CheckBox1]</p>	<p>➤ <i>ToolBox</i> panelini kullanabilirsiniz.</p>
<p>➤ <i>CheckBox</i> kontrolünün metnini <i>Deneme</i> olarak ayarlayın.</p> <p><input type="checkbox"/> Deneme</p>	<p>➤ <i>Properties</i> panelinden <i>Text</i> özelliğini kullanabilirsiniz.</p>
<p>➤ Uygulamayı çalıştırın.</p> <p><input type="checkbox"/> Deneme</p>	<p>➤ <i>Debug > Start Debugging (F5)</i> komutunu kullanabilirsiniz.</p>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Web sitesine bir tema ekleyebildiniz mi?		
2. Temaya bir dış görünümü dosyası ekleyebildiniz mi?		
3. Dış görünüm dosyasını kodlayabildiniz mi?		
4. Temayı sayfaya uygulayabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda “**Hayır**” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “**Evet**” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. () Temalar genel olarak web sitesinin nasıl olacağını belirleyen yapılardır.
2. () *App_Themes* ASP.NET'te temaları saklamak için kullanılan özel klasördür.
3. () Dış görünüm dosyalarının uzantısı *.skin*'dir.
4. () *SkinID* özelliğinin kullanılması zorunludur.
5. () Dış görünüm dosyaları kodlanırken kod tamamlama (intellisense) özelliğinden faydalanabilir.
6. () Sayfalara temalar *@Page* direktifi içine yazılan *Theme* özelliği ile eklenir.
7. () Bir kontrole uygulanmış temayı iptal etmek için *EnabledTheming* özelliği *True* olarak ayarlanmalıdır.
8. () Siteye tema *Web.config* dosyasına yazılan kodlarla eklenebilir.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-4

AMAÇ

Site haritası ve web sayfalarına menü ekleyerek site kullanımını kolaylaştırabileceksiniz.

ARAŞTIRMA

- Site haritasını araştırınız.
- Çeşitli web sitelerinde kullanılan menüleri inceleyiniz.

4. SAYFA YÖNLENDİRME

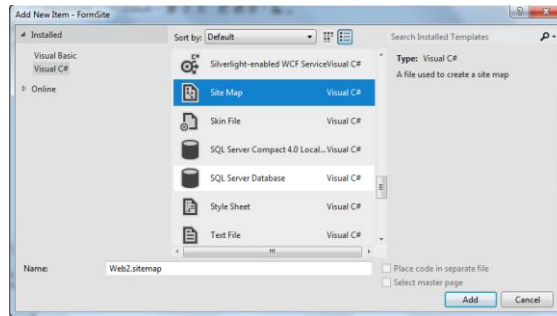
Bir web sayfası birden fazla sayfadan oluşmaktadır. Bu sayfalar arasında siteyi ziyaret eden kullanıcıların kolaylıkla dolaşabilmesi ve sitenin hiyerarşik yapısı hakkında bilgi sahibi olması büyük önem taşır. Programlama yazılımı, geliştiricilere bunları kolaylıkla yapabilmeleri için çeşitli menüler, site haritası ve site haritası yolu gibi çeşitli özellikler sunmaktadır. Bu özellikler kolaylıkla web sitesine eklenerek sitenin daha etkin bir hale getirilmesi sağlanabilir.

4.1. Site Haritası

Site haritası, web sitesinin sayfalarının hiyerarşik bir yapıda gösterilmesi için kullanılır. Programlama yazılımı ile site haritası oluşturmak için *SiteMap* dosyaları kullanılır. Bu dosyanın uzantısı *.sitemap*'tir. SiteMap dosyası diğer kontroller tarafından veri kaynağı olarak kullanılır. *SiteMapDataSource* kontrolü ile *SiteMap* dosyasından veriler okunur ve bu bilgiler menü vb. kontrollere aktarılabilir.

Web sitesinde bir site haritası oluşturmak için;

- Add New Item –FormSite iletişim penceresinden SiteMap komutunu seçin.



Resim 4.1: SiteMap dosyasının eklenmesi

- *SiteMap* dosyası eklendikten sonra bu dosyada web sitesine ait sayfaların belirtilmesi gerekir. Aşağıdaki kodları inceleyin.

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
  <siteMapNode url="" title="" description="">
    <siteMapNode url="" title="" description="" />
    <siteMapNode url="" title="" description="" />
  </siteMapNode>
</siteMap>
```

- *SiteMap* dosyasına bir sayfa eklenecekse *sitemapNode* etiketi kullanılır. Eğer eklenen sayfa alt düzeyde bir sayfa ise `<siteMapNode>.....</siteMapNode>` etiketleri arasında belirtilir. Bir alt düzeyde sayfa yoksa `<siteMapNode />` etiketi ile sonlandırılır.

- *url* alanına *SiteMap* içerisinde görüntülenecek sayfaların yolunu girin.

```
url="default.aspx"
```

- *title* alanına sayfaların gösterileceği kontroldeki başlık bilgisini girin.

```
title="Anasayfa"
```

- *description* alanına ise sayfalar ile ilgili açıklamaları girin.

```
description="Anasayfa"
```

Sayfa sayısı az olan web siteleri için bir *SiteMap* dosyasında sayfaların gösterilmesinde herhangi bir problem olmayacaktır. Ancak sayfa sayısı çok fazla olan bir sitede karmaşıklığa yol açacaktır. Bu durumda *SiteMap* içindeki veriler hiyerarşik yapıya göre farklı *SiteMap* dosyalarında tutularak *web.sitemap* dosyasında birleştirilebilir. Böylece karmaşıklığın önüne geçilmiş olur. *web.sitemap* dosyasının kodları aşağıdaki gibi düzenlenmelidir.

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
  <siteMapNode url="default.aspx" title="Anasayfa" description="Anasayfa">
    <siteMapNode siteMapFile="SiteMap1.sitemap"/>
    <siteMapNode siteMapFile="SiteMap2.sitemap"/>
  </siteMapNode>
</siteMap>
```

SiteMap dosyası hazırlandıktan sonra bu dosyadaki değerlerin bir kontrole aktarılması gerekir. Bir *SiteMap* dosyasından verileri okumak ve başka bir kontrole aktarmak için *SiteMapDataSource* kontrolü kullanılır. *SiteMapDataSource* kontrolü, yol belirtilmeksizin kök klasörü içinde *web.sitemap* dosyasını arar ve otomatik olarak okur.

SiteMapDataSource kontrolü *Toolbox* panelinde *Data* başlığı altında yer almaktadır. Sayfaya buradan eklenebilir.

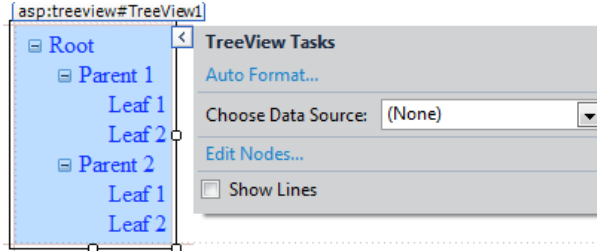
SiteMapDataSource kontrolü tek başına bir işlev görmeyecektir. *SiteMap* dosyası içindeki verileri sadece okuma işlemi yapar. Okuduğu bu verilerin menü vb. bir kontrole aktarılması gerekir. Bunun için;

- Boş bir sayfa ekleyin.
- *SiteMapDataSource* kontrolünü sayfaya ekleyin.

SiteMapDataSource - SiteMapDataSource1

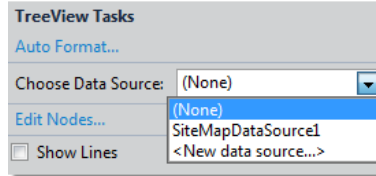
Resim 4.2: SiteMapDataSource kontrolü

- *Toolbox* paneli *Navigation* başlığı altından *TreeView* kontrolünü sayfaya ekleyin.



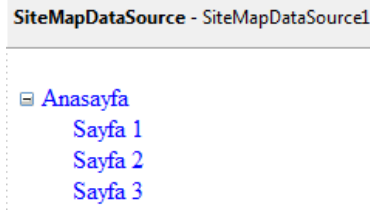
Resim 4.3: TreeView kontrolü

- *TreeView* kontrolü *Choose Data Source* özelliğinden oluşturduğunuz *SiteMapDataSource* kontrolünü gösterin.



Resim 4.4: SiteMapDataSource kontrolünün verilerinin aktarılması

- Aktarma işleminden sonra *SiteMap* dosyasında belirlenen sayfalar *TreeView* kontrolünde gösterilecektir.



Resim 4.5: Oluşturulmuş site haritası

4.2. Menüler

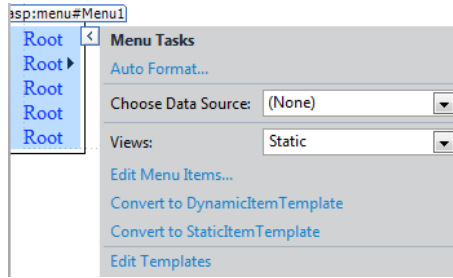
Programlama yazılımı geliştiricilerin kolaylıkla menü hazırlayabilmeleri için *Toolbox* panelinde *Navigation* başlığı altında çeşitli kontroller sağlamaktadır. Bu kontrollere veriler statik olarak eklenebileceği gibi çalışma zamanında da dinamik olarak eklenebilir.

4.2.1. Açılır ve Efektli Menüler

Toolbox paneli *Navigation* başlığı altında yer alan *Menu* kontrolü sayfaya eklenerek yatay veya dikey görünümde açılır menüler eklenebilir. Bu menüler JavaScript kod kullanılarak programlama yazılımında otomatik olarak oluşturulur. JavaScript kod bilmeye gerek yoktur.

Sayfa açılır menü eklemek için;

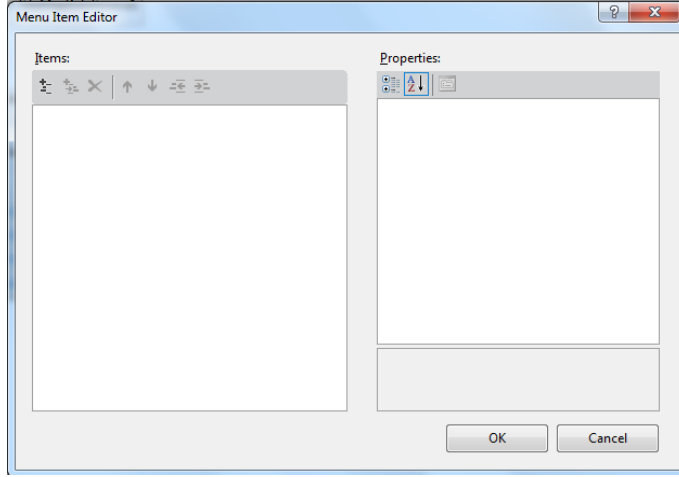
- *Toolbox* paneli *Navigation* başlığı altında *Menu* kontrolünü sayfaya ekleyin.



Resim 4.6: Menu kontrolü

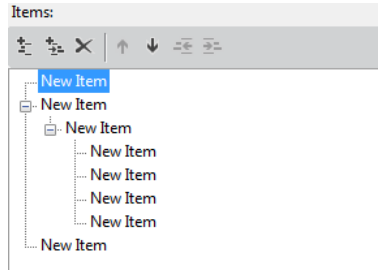
- *Menu* kontrolünün görev menüsünden *Edit Menu Items* komutunu seçin.
- Ekrana gelen *Menu Item Editor* iletişim penceresinden;
 - *Add a root item* komutu ile menüye ana menü başlığı ekleyebilirsiniz.
 - *Add a child item* komutu ile menüye alt menü başlığı ekleyebilirsiniz.
 - *Remove an item* komutu ile menü öğelerini silebilirsiniz.

- Bunların dışında menü öğelerinin yerlerini oklarla değiştirebilir, menü başlıklarının seviyesini bu iletişim penceresinden değiştirebilirsiniz.



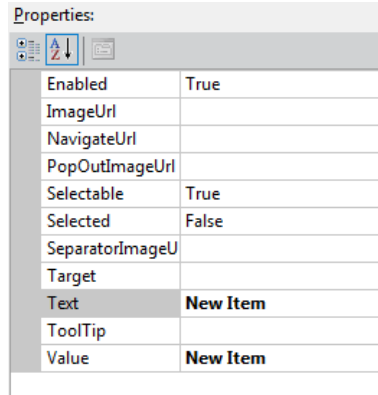
Resim 4.7: Menu Item Editor iletişim penceresi

- Web sitesinde kullanmayı düşündüğünüz menü öğelerini ekleyin. Çalışmanın ilerleyen bölümlerinde de istenildiği zaman menüler düzenlenebilir.



Resim 4.8: Menu kontrolüne öğe eklenmesi

- Her öğenin özelliklerinin ayarlanması gerekmektedir. Düzenlemek istediğiniz menü öğesini seçin ve iletişim penceresinin sağ tarafındaki özellikleri ayarlayın.



Resim 4.9: Menü öğesi özellikleri

- *Text* özelliği menü öğesinin metnini belirtir.
- *NavigateUrl* özelliği menü öğesi tıklandığında açılacak olan sayfayı belirtir.
- *Target* özelliği ise açılacak sayfanın hedef penceresini belirtir.

```

Anasayfa
Alanlar  Bilişim TeknolojileriAğ İşletmenliği
İletişim      Bilgisayar Teknik Servisi
              Veritabanı Programcılığı
              Web Programcılığı

```

Resim 4.10: Açılır menü

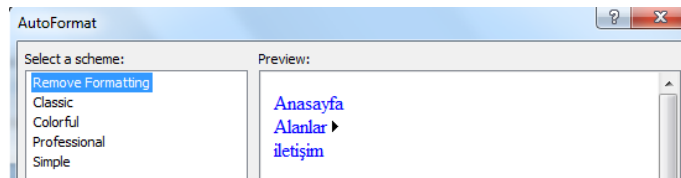
- İstendiğinde diğer kontrollerde olduğu gibi kontrol kodlarından da menü üzerinde değişiklik yapılabilir.

```

<asp:Menu ID="Menu1" runat="server">
  <Items>
    <asp:MenuItem Text="Anasayfa" Value="anasayfa"></asp:MenuItem>
    <asp:MenuItem Text="Alanlar" Value="alan">
      <asp:MenuItem Text="Bilişim Teknolojileri" Value="bilisim">
        <asp:MenuItem Text="Ağ İşletmenliği" Value="ag"></asp:MenuItem>
        <asp:MenuItem Text="Bilgisayar Teknik Servisi" Value="teknik">
          </asp:MenuItem>
        <asp:MenuItem Text="Veritabanı Programcılığı" Value="vertabani">
          </asp:MenuItem>
        <asp:MenuItem Text="Web Programcılığı" Value="web"></asp:MenuItem>
      </asp:MenuItem>
    <asp:MenuItem Text="iletişim" Value="iletisim"></asp:MenuItem>
  </Items>
</asp:Menu>

```

- Görev menüsünde *AutoFormat* özelliği ile programlama yazılımı menü şablonları kullanılabilir.



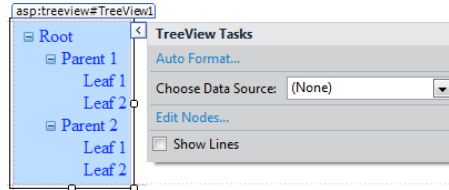
Resim 4.11: Menu kontrollünün görünümünün değiştirilmesi

4.2.2. Ağaç Görünümlü Menüler

Programlama yazılımında ağaç görünümlü menüler oluşturmak için *Toolbox* paneli *Navigation* başlığı altındaki *TreeView* kontrolü kullanılır. *Menu* kontrolünde olduğu gibi *TreeView* kontrolüne de öge eklenebilir. Farklı olarak *Items* yerine *TreeView* kontrolünde *Nodes* kullanılmaktadır. *TreeView* kontrolüne öge eklemek için görev menüsünden *Edit Nodes* özelliği kullanılır. *TreeView Node Editor* iletişim penceresinden *TreeView* kontrolüne *Menu* kontrolündeki gibi öge eklenebilir.

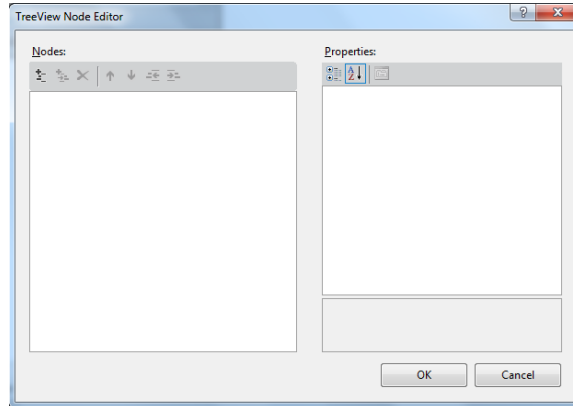
Sayfa ağaç görünümlü menü eklemek için;

- *Toolbox* paneli *Navigation* başlığı altında *TreeView* kontrolünü sayfaya ekleyin.



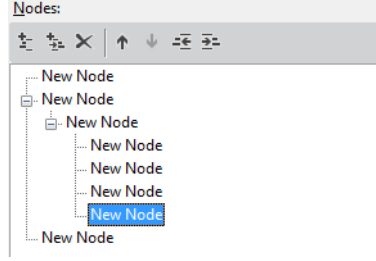
Resim 4.12: TreeView kontrolü

- *TreeView* kontrolünün görev menüsünden *Edit Nodes* komutunu seçin.
- Ekranaya gelen *TreeView Node Editor* iletişim penceresinden;
 - *Add a root node* komutu ile menüye ana menü başlığı ekleyebilirsiniz.
 - *Add a child node* komutu ile menüye alt menü başlığı ekleyebilirsiniz.
 - *Remove an node* komutu ile menü öğelerini silebilirsiniz.
 - Bunların dışında menü öğelerinin yerlerini oklarla değiştirebilir, menü başlıklarının seviyesini bu iletişim penceresinden değiştirebilirsiniz.



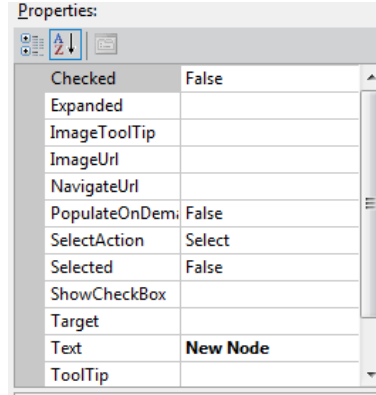
Resim 4.13: TreeView Node Editor iletişim penceresi

- Web sitesinde kullanmayı düşündüğünüz menü öğelerini ekleyin. Çalışmanın ilerleyen bölümlerinde de istenildiği zaman menüler düzenlenebilir.



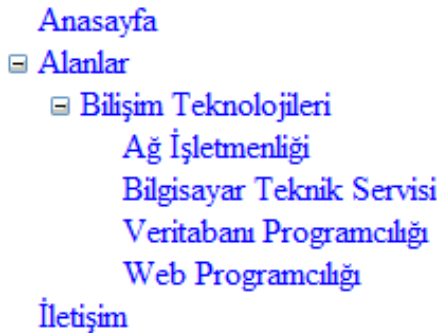
Resim 4.14: TreeView kontrolüne öğe eklenmesi

- Her öğenin özelliklerinin ayarlanması gerekmektedir. Düzenlemek istediğiniz menü öğesini seçin ve iletişim penceresinin sağ tarafındaki özellikleri ayarlayın.



Resim 4.15: TreeView öğesi özellikleri

- *Text* özelliği menü öğesinin metnini belirtir.
- *NavigateUrl* özelliği menü öğesi tıklandığında açılacak olan sayfayı belirtir.
- *Target* özelliği ise açılacak sayfanın hedef penceresini belirtir.

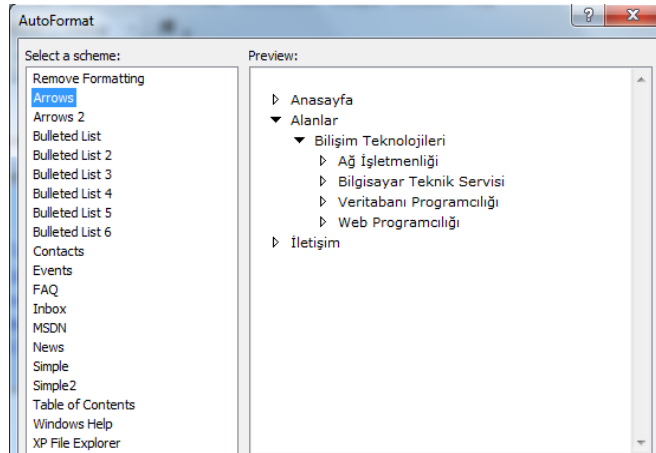


Resim 4.16: Açılır menü

- İstendiğinde diğer kontrollerde olduğu gibi kontrol kodlarından da ağaç görünümü menü üzerinde değişiklik yapılabilir.

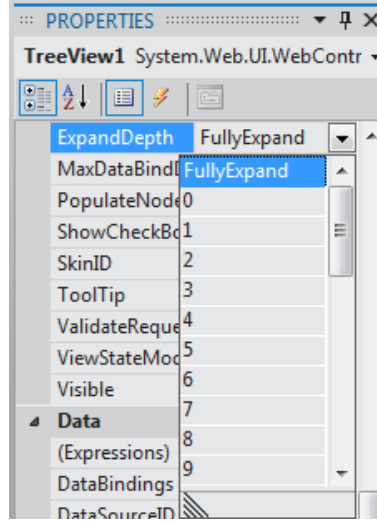
```
<asp:TreeView ID="TreeView1" runat="server">
  <Nodes>
    <asp:TreeNode Text="Anasayfa" Value="Anasayfa"></asp:TreeNode>
    <asp:TreeNode Text="Alanlar" Value="Alanlar">
      <asp:TreeNode Text="Bilişim Teknolojileri" Value="Bilişim Teknolojileri">
        <asp:TreeNode Text="Ağ İşletmenliği" Value="Ağ İşletmenliği">
        </asp:TreeNode>
        <asp:TreeNode Text="Bilgisayar Teknik Servisi"
          Value="Bilgisayar Teknik Servisi"></asp:TreeNode>
        <asp:TreeNode Text="Veritabanı Programcılığı" Value="Veritabanı
          Programcılığı">
        </asp:TreeNode>
        <asp:TreeNode Text="Web Programcılığı" Value="Web
          Programcılığı"></asp:TreeNode>
      </asp:TreeNode>
    </asp:TreeNode>
    <asp:TreeNode Text="İletişim" Value="İletişim"></asp:TreeNode>
  </Nodes>
</asp:TreeView>
```

- Görev menüsünde *AutoFormat* özelliği ile programlama yazılımı *TreeView* şablonları kullanılabilir.



Resim 4.17: TreeView kontrollünün görünümünün değiştirilmesi

- *TreeView* kontrolü içeren bir sayfa açıldığında varsayılan olarak tüm seviyeler açılır. Ancak istendiğinde sayfa açıldığında hangi seviyelerin açılacağı ayarlanabilir. Bunun için *TreeView* kontrolü *ExpandDepth* özelliğinden seviye 0'dan başlayarak belirlenebilir. *FullyExpand* olarak seçilirse tüm seviyeler gösterilir.



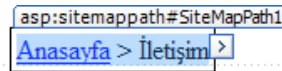
Resim 4.18: Treview kontrolü gösterilecek seviyelerin belirlenmesi

4.2.3. Site Haritası Yolu

Kullanıcılar sitede dolaşırken bulunduğu sayfa hiyerarşik olarak gösterilebilir. Böylece üst seviyelere rahatlıkla geri dönülmesi sağlanabilir. Programlama yazılımı bunu gerçekleştirmek için *SiteMapPath* kontrolüne sahiptir. *SiteMapPath* kontrolü sayfaya eklendikten sonra *SiteMap* dosyası içerisindeki bilgileri otomatik olarak okur.

Site Haritası Yolu eklemek için;

- *Toolbox* paneli *Navigation* başlığı altında *SiteMapPath* kontrolünü sayfaya ekleyin.



Resim 4.19: SiteMapPath kontrolü

- *SiteMapPath* kontrolü *SiteMap* dosyasından verileri otomatik olarak okuyarak eklenen sayfanın hiyerarşik olarak yerini gösterecektir.

UYGULAMA FAALİYETİ

Aşağıdaki işlem basamaklarını takip ederek ağaç görünümlü menü kullanarak bir *sitemap* oluşturunuz.

İşlem Basamakları	Öneriler
<p>➤ Yeni bir boş web projesi oluşturun.</p>	<p>➤ <i>File > New > Web Site</i> komutunu kullanabilirsiniz.</p> <p>➤ <i>New WebSite</i> iletişim penceresinden <i>ASP.NET Empty Web Site</i> komutunu kullanabilirsiniz.</p>
<p>➤ Web sitesine dört tane web sayfası ekleyin.</p>	<p>➤ <i>Website > Add New Item</i> iletişim penceresini kullanabilirsiniz.</p>
<p>➤ Web sitesine <i>SiteMap</i> dosyası ekleyin.</p>	<p>➤ <i>Website > Add New Item</i> iletişim penceresini kullanabilirsiniz.</p>
<p>➤ <i>SiteMap</i> dosyası kodlarına sayfaları ekleyin.</p> <pre><?xml version="1.0" encoding="utf-8" ?> <siteMap xmlns="http://schemas.microsoft.com/ AspNet/SiteMap-File-1.0" > <siteMapNode url="" title="Site Haritası" description=""> <siteMapNode url="default.aspx" title="Sayfa 1" description="Sayfa1" /> <siteMapNode url="default2.aspx" title="Sayfa 2" description="Sayfa2" /> <siteMapNode url="default3.aspx" title="Sayfa 3" description="Sayfa3" /> <siteMapNode url="default4.aspx" title="Site Haritası" description="Sayfa4" /> </siteMapNode> </siteMap></pre>	
<p>➤ <i>Sayfa 4'e</i> gelin ve <i>SiteMap</i> dosyasındaki verileri okumak için sayfaya <i>SiteMapDataSource</i> kontrolü ekleyin.</p> <div style="border: 1px solid gray; padding: 2px; width: fit-content;">SiteMapDataSource - SiteMapDataSource1</div>	<p>➤ <i>ToolBox</i> panelini kullanabilirsiniz.</p>

<p>➤ <i>Sayfa 4'e SiteMap</i> dosyasındaki verilerin gösterilmesi için <i>TreeView</i> kontrolü ekleyin.</p> <ul style="list-style-type: none"> ▣ Root <ul style="list-style-type: none"> ▣ Parent 1 <ul style="list-style-type: none"> Leaf 1 Leaf 2 ▣ Parent 2 <ul style="list-style-type: none"> Leaf 1 Leaf 2 	<p>➤ <i>ToolBox</i> panelini kullanabilirsiniz.</p>
<p>➤ <i>TreeView</i> kontrolü görünümü değiştirin.</p>	<p>➤ <i>TreeView</i> kontrolü görev menüsünden <i>Auto Format</i> özelliğini kullanabilirsiniz.</p>
<p>➤ <i>TreeView</i> kontrolüne <i>SiteMapDataSource</i> kontrolü verileri aktarın.</p>	<p>➤ <i>TreeView</i> kontrolü görev menüsünden <i>Choose Data Source</i> özelliğini kullanabilirsiniz.</p>
<p>➤ <i>Sayfa 4'ü</i> tarayıcıda görüntüleyin.</p> <ul style="list-style-type: none"> ▼ Site Haritası <ul style="list-style-type: none"> ▸ Sayfa 1 ▸ Sayfa 2 ▸ Sayfa 3 ▸ Site Haritası 	<p>➤ <i>Debug > Start Debugging (F5)</i> komutunu kullanabilirsiniz.</p>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Yeni proje oluşturabildiniz mi?		
2. <i>SiteMap</i> dosyası oluşturabildiniz mi?		
3. <i>SiteMap</i> dosyasında sayfaları tanımlayabildiniz mi?		
4. <i>SiteMap</i> dosyası verilerini okumak için <i>SiteMapDataSource</i> kontrolünü ekleyebildiniz mi?		
5. Site haritası için menü ekleyebildiniz mi?		
6. Menünün görünümünü değiştirebildiniz mi?		
7. Menüye <i>SiteMapDataSource</i> verilerini aktarabildiniz mi?		
8. <i>SiteMap</i> sayfasını çalıştırabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda “**Hayır**” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “**Evet**” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. () Site haritası web sitesinin sayfalarının hiyerarşik bir yapıda görüntülenmesini sağlar.
2. () Bir web sayfasında sadece bir tane *SiteMap* dosyası kullanılabilir.
3. () *SiteMapDataSource* kontrolü *SiteMap* dosyasından verileri okumak için kullanılır.
4. () *SiteMap* verileri bir menü kontrolüne aktarılabilir.
5. () Açılır menülerin öğeleri sadece statik olarak atanabilir.
6. () Ağaç görünümlü menülerin görünümleri değiştirilemez.
7. () Site haritası yolu için *SiteMapPath* kontrolü kullanılır.
8. () *SiteMapPath* kontrolü verileri *SiteMap* dosyasını otomatik olarak çeker.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-5

AMAÇ

Site ve web sayfaları üzerinde durum yönetimi sağlayabileceksiniz.

ARAŞTIRMA

- Durum yöneticilerine neden ihtiyaç duyulduğunu araştırınız.

5. DURUM YÖNETİCİSİ

Web sayfaları kullanılırken istemci ve sunucu arasındaki bilgi alış verişi HTTP protokolü aracılığıyla gerçekleştirilmektedir. Geleneksel web programlamada, istemci ile sunucu arasındaki gerçekleşen bilgi alışverişinde, istemciden sunucuya iletilen bilgiler kalıcı değildir ve tekrar erişilemez. Bilgiler kullanıldıktan sonra sunucu belleğinden kaldırılır.

Geleneksel web programlamada bu sınırlamayı aşmak için ASP.NET içerisinde istemci bilgilerine tekrar ulaşılabilmesi, bu bilgilerin saklanması ve taşınması için bazı yapılar oluşturulmuştur.

Web uygulamalarında veriler için durum yönetimi sağlayarak verilerin sunucu ya da istemci tarafından saklanmasını sağlayan yapılara *durum yönetimi nesnelere* adı verilmektedir. Bunlar;

- Sorgulama Cümlesi (Query String)
- Görünüm Durumu Yönetimi (View State Management)
- Çerezler (Cookies)
- Oturum Yönetimi (Session Management)
- Uygulama Durum Yönetimi (Application State Management)

5.1. Sorgulama Cümlesi (QueryString)

QueryString nesnesi ile sayfalar arasında taşınacak veri direk olarak URL aracılığıyla taşınır. Taşınacak olan veriler sayfanın adresinden sonra soru işareti (?) başlayan kısımda taşınır. Bu yöntem hem kullanım kolaylığı hem de sunucuya getirdiği yükün az olması nedeniyle çokça tercih edilmektedir.

/custom?q=3308&sitesearch=mevzuat.meb.gov.tr&client=pub-&foi

Resim 5.1: QueryString ile veri aktarımı

Resimdeki URL adresi incelenecek olursa q=3308 şeklinde sözcük belirtilmiştir. Aramayı daha da detaylandırmak için & işareti ile anahtar kelimeler kullanılmıştır.

QueryString ile taşınan veriler kullanıcılar tarafından görüntülenebilmektedir. Bu durum güvenlik açığı oluşturmaktadır. Veriler şifreli gönderilse bile şifreli metinler kullanıcılar tarafından görülmesi güvenlik açığını % 100 ortadan kaldırmamaktadır. Bu nedenle QueryString ile güvenliği önemli bilgiler taşınmamalıdır.

QueryString ile veri aktarımı yapmak için;

- Boş bir sayfa açın ve aşağıdaki kontrolleri ekleyin.

Resim 5.2: QueryString ile verilerin kontrollerle aktarılması

- Button *Click* olayına aşağıdaki kodları ekleyin.

```
Response.Redirect("default2.aspx?Goruntule="+TextBox1.Text+"");
```

- İkinci bir sayfa ekleyin ve *Page_Load* metoduna aşağıdaki kodları ekleyin.

```
string yaz = Request.QueryString["Goruntule"];  
Response.Write(yaz);
```

- Uygulamayı çalıştırın ve metin kutusuna bir değer girin.

Resim 5.3: QueryString ile aktarılabacak verinin girilmesi

- Button tıklandıktan sonra ikinci sayfanın açıldığında sayfa adresinde QueryString ile girilen değer aktarıldığı görülebilir.

http://localhost:1335/default5.aspx?Goruntule=MTEGM

Resim 5.4: QueryString ile veri aktarılan sayfa adresi

- Açılan ikinci sayfada aktarılan veri görülebilir.

MTEGM

Resim 5.5: QueryString ile aktarılan veri

5.2. Görünüm Durumu Yönetimi (ViewState)

ASP.NET'te form düzeyinde verileri saklamak için kullanılan *ViewState* nesnesi, sayfa içindeki kontrol özelliklerini ve geliştiricilerin istediği verileri saklar. Bu veriler *PostBack* işlemi gerçekleştiğinde şifrelenmiş bir şekilde *ViewState* içerisine yazılır. Sayfa tekrar yüklendiğinde kontrol özellikleri *ViewState* nesnesinden okunur.

ViewState nesnesi ile bir kontroldeki veri aktarılması için kontrolün *EnableViewState* özelliğinin *True* olarak ayarlanması gerekmektedir. Varsayılan olarak bu özellik *True* değerdedir.

Kontrollerin sadece *Text* özelliği değil tüm özellikleri (Width, Height vb.) *ViewState* nesnesi ile taşınmaktadır.

ViewState nesnesi ile verileri aktarmak için;

- Yeni bir sayfa açın.
- Aşağıdaki kontrolleri sayfaya ekleyin.

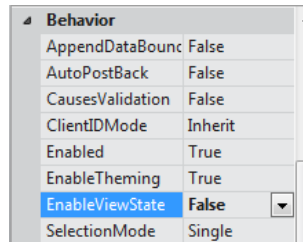


Resim 5.6: ViewState ile kontrol özelliklerinin aktarılması

- Button *Click* olayına aşağıdaki kodları ekleyin.

```
ListBox1.Items.Add(TextBox1.Text);
```

- ListBox kontrolünün *EnableViewState* özelliğini *False* olarak ayarlayın. Varsayılan değeri *True* değerdedir.



Resim 5.7: EnableViewState özelliği

- Uygulamayı çalıştırın ve *Ekle* butonu aracılığıyla *TextBox* içindeki metni *ListBox* kontrolüne aktarın. Bu işlemi birkaç defa tekrarlayın.

- Her eklemede *ListBox* kontrolündeki değerler *PostBack* işleminden sonra aktarılmayacak ve yeni değer *ListBox* kontrolüne eklenecektir.

Resim 5.8: EnableViewState özelliği false değerde iken ekran çıktısı

- *ListBox* kontrolünün *EnableViewState* özelliğini *True* olarak ayarlayın.
- Uygulamayı çalıştırın ve *Ekle* butonu aracılığıyla *TextBox* içindeki metni *ListBox* kontrolüne aktarın. Bu işlemi birkaç defa tekrarlayın.
- Her eklemede *ListBox* kontrolündeki değerler *PostBack* işleminden sonra korunacak ve yeni değer *ListBox* kontrolüne eklenecektir.

Resim 5.9: EnableViewState özelliği true değerde iken ekran çıktısı

5.3. Çerezler (Cookies)

Web uygulamalarında verilerin başka bir aktarılma yöntemi ise verilerin kullanıcıların bilgisayarında fiziksel olarak saklanmasıdır. *Cookie* nesnesi istemci bilgisayarında verileri fiziksel olarak saklamak için kullanılır. Fiziksel ortamda saklandığı için istenilen zamanda verilere erişilebilir. Ancak bir dezavantajı vardır; kullanıcılar *Cookie* kaydını engelledikleri zaman kullanılamayabilir.

Cookie nesnesi ile ilgili işlemler yapılabilmesi için ASP.NET uygulamalarında *HttpCookies* sınıfı kullanılmaktadır. *HttpCookies* sınıfından oluşturulan nesne ile *Cookie* bilgilerinin yazılması, okunması ve bilgilerin saklanacağı süre işlemleri yapılabilir.

Bilgileri *HttpCookies* kullanarak kaydetmek için;

- Yeni bir sayfa açın.
- Aşağıdaki kontrolleri sayfaya ekleyin.

Resim 5.10: Cookie nesnesi uygulaması

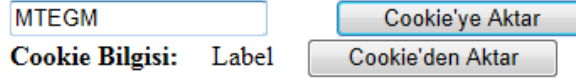
- *Cookie'ye Aktar* butonuna aşağıdaki kodları ekleyin.

```
HttpCookie cerez = new HttpCookie("CookieAktar");  
cerrez["bilgi"] = TextBox1.Text;  
cerrez.Expires = DateTime.Now.AddDays(10);  
Response.Cookies.Add(cerez);
```

- *Cookie'den Aktar* butonuna aşağıdaki kodları ekleyin.

```
HttpCookie cerezaktar = Request.Cookies["CookieAktar"];  
Label2.Text = cerezaktar["bilgi"];
```

- Uygulamayı çalıştırın ve Cookie değerini aktarın.



Resim 5.11: Bilgilerin Cookie'ye aktarılması

- *Cookie'den Aktar* butonu ile bilgiyi okutun.



Resim 5.12: Bilgilerin Cookie'den aktarılması

5.4. Oturum Yönetimi (Session Management)

Web sunucusu, kendisinden bir sayfa talep edildiğinde her kullanıcı için bir oturum başlatır. Oluşturulan kullanıcı oturumu boyunca *Session* nesnesi ile sayfalar arasında bilgi taşıma işlemi yapılabilir. *Session* nesnesi ile saklanan bilgilere oturum açık olduğu sürece her sayfadan ulaşılabilir, oturum kapatıldığı anda yok edilir.

Oturum açan her kullanıcı için bir *SessionID* değeri üretilir. Bu değer hem sunucuda hem de istemci tarafında saklanır. Sunucu istek gelen istemciyi *SessionID* değerlerini karşılaştırarak belirler. *SessionID* değeri istemci tarafında *Cookie*'ler aracılığıyla saklanır. *Cookie* istemcide aktif değilse URL ile birlikte saklanabilmektedir.

Session nesnesi sunucuda çalışan bir nesne olduğu için *SessionID* değeri haricinde istemciye bir bilgi gönderilmez, bilgiler sunucuda saklanır. Sunucu belleğinin verimli kullanılabilmesi sadece önemli bilgilerin *Session* nesnesi ile taşınması, önemsiz bilgilerin ise *QueryString* nesnesi ile taşınması uygun olacaktır.

Session nesnesinin kullanımı için;

- Yeni bir *MasterPage* ve bu sayfadan iki tane türemiş sayfa oluşturun.
- *MasterPage* sayfasına aşağıdaki kontrolleri ekleyin. (Sayfalar arası geçiş için *SiteMapPath - Site Haritası Yolu* eklenmiştir.)

Hoşgeldin,Label Oturumu Kapat
Kullanıcı Adı
Şifre
Giriş

Root Node : Parent Node : Current Node

Resim 5.13: MasterPage sayfası tasarımı

- *Sayfa 1* ve *Sayfa 2*'yi ayırt etmek için bu sayfalara bilgiler girin.

Hoşgeldin,Label Oturumu Kapat
Kullanıcı Adı
Şifre
Giriş
Sayfa1
ContentPlaceHolder1 (Custom) p.auto-style4
Sayfa 1 görüntüleniyor!

Hoşgeldin,Label Oturumu Kapat
Kullanıcı Adı
Şifre
Giriş
Sayfa1 : Sayfa2
ContentPlaceHolder1 (Custom) p.auto-style4
Sayfa 2 görüntüleniyor!

Resim 5.14: Türemiş sayfaların tasarımı

- *MasterPage* sayfası *Page_Load* olayına aşağıdaki kodları ekleyin.

```
if (Session["KullaniciAdi"] == null)
{
    Label1.Text = "Ziyaretçi!";
    Button2.Visible = false;
}
else
{
    Label1.Text = Session["KullaniciAdi"].ToString();
    Label2.Visible = false;
    Label3.Visible = false;
    TextBox1.Visible = false;
    TextBox2.Visible = false;
    Button1.Visible = false;
    Button2.Visible = true;
}
```

- Bu kodlar ile *Session* nesnesi ile bir bilgi taşınıp taşınmadığı kontrol edilmiştir. Bilgi taşınmıyorsa kullanıcı adı *Ziyaretçi* olarak girilmektedir. Bir değer taşınıyorsa kullanıcı adı yazılmış ve giriş paneli gizlenmiştir.
- *Giriş* butonu *Click* olayına aşağıdaki kodları ekleyin.

```
if (TextBox1.Text == "Admin" && TextBox2.Text == "1234")
{
    Session.Add("KullaniciAdi", TextBox1.Text);
    Label1.Text = Session["KullaniciAdi"].ToString();
    Response.Redirect("Default2.aspx");
}
```

- Bu kodlar ile kullanıcı adı ve şifre kontrol edilmiş ve kullanıcı adı *Session* nesnesine aktarılmıştır. Kullanıcı adı aktarılmış ve sayfa 2'ye yönlendirilmiştir.
- *Sayfa 2*'nin oturum açılmadan görüntülenmesini engellemek için *page_Load* olayına aşağıdaki kodları yazın.

```
if (Session["KullaniciAdi"] == null)
{
    Response.Redirect("default.aspx");
}
```

- *Oturumu kapat* butonu tıklandığında giriş sayfasına geçiş yapmak için *Click* olayına aşağıdaki kodları yazın.

```
Session.Abandon();
Response.Redirect("Default.aspx");
```

- Uygulamayı çalıştırın ve giriş yapın.

The image shows three stages of a web application's login process:

- Stage 1:** The user is on the login page. The text "Hoşgeldin, Ziyaretçi!" is displayed. There are two input fields: "Kullanıcı Adı" (User Name) and "Şifre" (Password). A "Giriş" (Login) button is at the bottom. Below the form, it says "Sayfa 1 görüntüleniyor!" (Page 1 is being displayed!).
- Stage 2:** The user has entered "Admin" in the "Kullanıcı Adı" field and "1234" in the "Şifre" field. The "Giriş" button is still visible.
- Stage 3:** The user is now logged in. The text "Hoşgeldin,Admin" is displayed. There is a "Oturumu Kapat" (Logout) button. Below the text, it says "Sayfa 1 : Sayfa2" and "Sayfa 2 görüntüleniyor!" (Page 2 is being displayed!).

Resim 5.15: Sayfa görüntüleri

- Oturumu kapattığınızda sayfa ilk haline geri dönecektir.

5.5. Uygulama Durum Yönetimi (Application State Management)

Application nesnesi ile web sitesinin tamamını ilgilendiren bilgiler tutulur. Siteye erişen her kullanıcı *Application* nesnesi ile tutulan bilgileri görebilir, yani bu nesne ile tutulan her bilgi tüm kullanıcılar için ortaktır.

Application nesnesinin kullanımına örnek olarak;

- Bir önceki uygulamayı çalıştırın.
- *MasterPage* sayfasının tasarımını aşağıdaki gibi değiştirin.

The image shows a web page design for a MasterPage. It features a login form with the following elements:

- A label "Hoşgeldin, Label" above a text input field.
- A label "Kullanıcı Adı" above a text input field.
- A label "Şifre" above a text input field.
- A "Giriş" button below the password field.
- An "Oturumu Kapat" button to the right of the "Hoşgeldin, Label" field.

Below the form, there is a status bar with the text "Root Node : Parent Node : Current Node" and "Online Kullanıcı Sayısı: Label".

Resim 5.16: MasterPage sayfası tasarımı

- Online kullanıcı sayısını kontrol edilebilmesi uygulamaya *Global.asax* dosyasının eklenmesi gerekmektedir. Uygulamada bu dosya ekli değilse *Add New Item* iletişim penceresinden *Global Application Class* komutu seçilmelidir.
- *void Application_Start* metodu altına aşağıdaki kodları ekleyin. Bu kod uygulama başlatıldığında kullanıcı sayısını sıfırlayacaktır.

```
Application["KS"] = 0;
```

- *void Session_Start* metodu altına aşağıdaki kodları ekleyin. Bu kod uygulamada oturum açıldığında kullanıcı sayısını artıracaktır.

```
Application["KS"] = ((int)Application["KS"]) + 1;
```

- *void Session_End* metodu altına aşağıdaki kodları ekleyin. Bu kod uygulamada oturum kapatıldığında kullanıcı sayısını azaltacaktır.

```
Application["KS"] = ((int)Application["KS"]) - 1;
```

- *MasterPage* sayfası *Page_Load* olayına aşağıdaki kodu ekleyin.

```
Label4.Text = Convert.ToString(Application["KS"]);
```

- Uygulamayı çalıştırın ve online kullanıcı sayısını gözlemleyin.

Hoşgeldin, Ziyaretçi!
Kullanıcı Adı
Şifre

Sayfa1


Sayfa 1 görüntüleniyor!

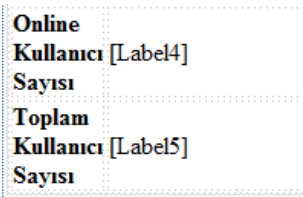
Online Kullanıcı Sayısı: 1

Resim 5.17: Online kullanıcı sayısı ekran çıktısı

UYGULAMA FAALİYETİ

Durum yönetimi nesnelere ile ilgili aşağıdaki uygulamayı yapınız.

İşlem Basamakları	Öneriler
<p>➤ Yeni bir boş web projesi oluşturun.</p>	<p>➤ <i>File > New > Web Site</i> komutunu kullanabilirsiniz.</p> <p>➤ <i>New WebSite</i> iletişim penceresinden <i>ASP.NET Empty Web Site</i> komutunu kullanabilirsiniz.</p>
<p>➤ Web sayfasını açın ve aşağıdaki tasarıma uygun kontrolleri sayfaya ekleyin.</p> <div data-bbox="308 768 611 1069" data-label="Form"></div>	<p>➤ <i>Toolbox</i> panelini kullanabilirsiniz.</p>
<p>➤ Oturum yönetimini kontrol etmek için sayfa <i>Page_Load</i> olayına aşağıdaki kodları yazın.</p> <pre>if (Session["KullaniciAdi"] == null) { Label3.Text = "Ziyaretçi!"; Button2.Visible = false; } else { Label3.Text = Session["KullaniciAdi"].ToString(); Label1.Visible = false; Label2.Visible = false; TextBox1.Visible = false; TextBox2.Visible = false; Button1.Visible = false; Button2.Visible = true; }</pre>	<p>➤ <i>Page_Load</i> olayı kod sayfasına otomatik olarak eklenmiş durumda olacaktır.</p>

<p>➤ Giriş butonu <i>Click</i> olayına aşağıdaki kodları ekleyin.</p> <pre> if (TextBox1.Text == "Admin" && TextBox2.Text == "1234") { Session.Add("KullaniciAdi", TextBox1.Text); Label3.Text = Session["KullaniciAdi"].ToString(); Label1.Visible = false; Label2.Visible = false; TextBox1.Visible = false; TextBox2.Visible = false; Button1.Visible = false; Button2.Visible = true; } </pre>	<p>➤ Buton üzerinde çift tıklayarak <i>Click</i> olayını ekleyebilirsiniz.</p>
<p>➤ Oturumu kapat butonu <i>Click</i> butonuna aşağıdaki kodları yazın.</p> <pre> Session.Abandon(); Response.Redirect("Default.aspx"); </pre>	<p>➤ Buton üzerinde çift tıklayarak <i>Click</i> olayını ekleyebilirsiniz.</p>
<p>➤ Giriş Paneli altına aşağıdaki kontrolleri ekleyin.</p> 	<p>➤ <i>Toolbox</i> panelini kullanabilirsiniz.</p>
<p>➤ Web sitesine <i>Global.asax</i> dosyası ekleyin.</p>	<p>➤ <i>Add New Item</i> iletişim penceresinden <i>Global Application Class</i> komutunu kullanabilirsiniz.</p>
<p>➤ <i>void Application_Start</i> metodu altına aşağıdaki kodları ekleyin.</p> <pre> Application["KS"] = 0; Application["TS"] = 0; </pre>	<p>➤ Bu kod ile uygulama başlatıldığında toplam ve online kullanıcı sayısını sıfırlayabilirsiniz.</p>

<p>➤ <code>void Session_Start</code> metodu altına aşağıdaki kodları ekleyin.</p> <pre>Application["KS"] = ((int) Application["KS"]) + 1; Application["TS"] = ((int) Application["TS"]) + 1;</pre>	<p>➤ Bu kod ile oturum açıldığında online kullanıcı sayısı ve toplam kullanıcı sayısını artırabilirsiniz.</p>
<p>➤ <code>void Session_End</code> metodu altına aşağıdaki kodları ekleyin.</p> <pre>Application["KS"] = ((int) Application["KS"]) - 1;</pre>	<p>➤ Bu kod uygulamada oturum kapatıldığında online kullanıcı sayısını azaltacaktır.</p>
<p>➤ Aşağıdaki kodları sayfanın <code>Page_Load</code> olayına yazınız.</p> <pre>Label4.Text = Convert.ToString(Application["KS"]); Label5.Text = Convert.ToString(Application["TS"]);</pre>	<p>➤ Bu kodlar ile online kullanıcı sayısı ve toplam kullanıcı sayılarını kontrollere aktarabilirsiniz.</p>
<p>➤ Uygulamayı çalıştırın.</p>	<p>➤ <code>Debug > Start Debugging (F5)</code> komutunu kullanabilirsiniz.</p>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Oturum yönetimi nesnesini kullanabildiniz mi?		
2. Oturum açma – kapatma işlemlerini gerçekleştirebildiniz mi?		
3. Uygulama durum yönetimi nesnesini kullanabildiniz mi?		
4. Uygulama durum yönetimi için <i>Global.asax</i> dosyasını kodlayabildiniz mi?		
5. Online ve toplam kullanıcı sayılarını okutabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda “**Hayır**” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “**Evet**” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. () *Durum yönetim nesneleri* verilen sunucu ya da istemci tarafında saklanması için kullanılır.
2. () *QueryString* nesnesi ile sayfalar arasında taşınacak veri direk olarak http aracılığıyla taşınır.
3. () *QueryString* taşınan veriler sayfa adresinde yer alan *soru işareti (?)* simgesinden sonra gösterilir.
4. () *ViewState* nesnesine veriler *PostBack* işleminden sonra şifrelenmiş bir şekilde yazılır.
5. () *Cookie* verileri sunucu üzerinde verileri fiziksel olarak saklamak için kullanılır.
6. () *Cookie* nesnesi ile ilgili işlemler yapılabilmesi için ASP.NET uygulamalarında *HttpCookies* sınıfı kullanılmaktadır.
7. () Oturum açan her kullanıcı için oluşturulan *SessionID* değeri tekrar tekrar kullanılabilir.
8. () *Application* nesnesi ile web sitesinin tamamını ilgilendiren bilgiler tutulur.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise “Modül Değerlendirme”ye geçiniz.

MODÜL DEĞERLENDİRME

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

1. Anasayfa dosyalarının uzantısı aşağıdakilerden hangisidir?
 - A) .aspx
 - B) .asax
 - C) .master
 - D) .config
2. Anasayfa dosyaları hangi direktif ile başlar?
 - A) @Page
 - B) @Master
 - C) @Control
 - D) @Register
3. Aşağıdakilerden hangisi *ContentPalceHolder* kontrolü için **yanlıştır**?
 - A) *MasterPage* sayfasından türeyen sayfalar için düzenlenebilir alanlardır.
 - B) *MasterPage* sayfasında birden fazla *ContentPlaceHolder* kontrolü kullanılabilir.
 - C) *MasterPage* sayfasında programlama yazılımı başlangıçta iki tane *ContentPlaceHolder* kontrolü ekler.
 - D) *MasterPage* sayfasında programlama yazılımı başlangıçta eklenen *ContentPlaceHolder* kontrolleri silinemez.
4. Stil sayfalarının dosya uzantısı aşağıdakilerden hangisidir?
 - A) .aspx
 - B) .css
 - C) .master
 - D) .config
5. Stil sayfası kuralları belirlenirken kuralın isminin önünde aşağıdaki noktalama işaretlerinden hangisi kullanılır?
 - A) Nokta (.)
 - B) Virgül (,)
 - C) Ünlem (!)
 - D) Noktalı Virgül (;)
6. Stil dosyaları sayfalara bağlanırken hangi etiketler arasında tanımlanır?
 - A) <body>...</body>
 - B) <title>...</title>
 - C) <head>...</head>
 - D) <form>...</form>

7. ASP.NET web sitelerinde temalar hangi klasör altında tutulur?
A) *App_Data*
B) *App_Themes*
C) *App_Code*
D) *App_Browser*
8. Dış görünüm dosyaları aşağıdakilerden hangisidir?
A) *Style Sheet*
B) *Skin File*
C) *SiteMap*
D) *Text File*
9. Web sitesine tema uygulamak için hangi dosya kodlarına ekleme yapılmalıdır?
A) *Global.asax*
B) *Web.sitemap*
C) *Favicon.ico*
D) *Web.config*
10. Çalışma zamanında sayfanın temasının belirlenmesi için gerekli kodlar hangi sayfa olayına yazılmalıdır?
A) *Page_Load*
B) *Page_PreRender*
C) *Page_Unload*
D) *Page_PreInit*
11. Site haritası dosyasının uzantısı aşağıdakilerden hangisidir?
A) *.css*
B) *.sitemap*
C) *.config*
D) *.asax*
12. Açılır menüler ile ilgili olarak aşağıdakilerden hangisi **yanlıştır**?
A) Bir sayfaya birden fazla açılır menü oluşturulabilir.
B) Açılır menülerin görünümleri değiştirilebilir.
C) Açılır menü öğeleri sadece statik olarak eklenebilir.
D) Açılır menü öğeleri seviyelere göre gruplanabilir.
13. *QueryString* taşınan veriler sayfa adresinde yer alan hangi simgeden sonra gösterilir?
A) ! (Ünlem)
B) ? (Soru İşareti)
C) * (Yıldız)
D) = (Eşittir)

14. Aşağıdaki durum yönetim nesnelereinden hangisi verileri istemci bilgisayarında fiziksel olarak saklamak için kullanılır?
- A) *QueryString*
 - B) *ViewState*
 - C) *Cookie*
 - D) *Session*
15. Aşağıdaki durum yönetim nesnelereinden hangisinde tutulan veriler *PostBack* işleminden sonra şifreli olarak gönderilir?
- A) *ViewState*
 - B) *Cookie*
 - C) *Application*
 - D) *QueryString*

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki modüle geçmek için öğretmenimize başvurunuz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ-1'İN CEVAP ANAHTARI

1	Doğru
2	Yanlış
3	Doğru
4	Yanlış
5	Yanlış
6	Doğru
7	Doğru
8	Doğru

ÖĞRENME FAALİYETİ-2'NİN CEVAP ANAHTARI

1	Doğru
2	Doğru
3	Yanlış
4	Doğru
5	Yanlış
6	Doğru

ÖĞRENME FAALİYETİ-3'ÜN CEVAP ANAHTARI

1	Doğru
2	Doğru
3	Doğru
4	Yanlış
5	Yanlış
6	Doğru
7	Yanlış
8	Doğru

ÖĞRENME FAALİYETİ-4'ÜN CEVAP ANAHTARI

1	Doğru
2	Yanlış
3	Doğru
4	Doğru
5	Yanlış
6	Yanlış
7	Doğru
8	Doğru

ÖĞRENME FAALİYETİ-5'İN CEVAP ANAHTARI

1	Doğru
2	Yanlış
3	Doğru
4	Doğru
5	Yanlış
6	Doğru
7	Yanlış
8	Doğru

MODÜL DEĞERLENDİRME'NİN CEVAP ANAHTARI

1	C
2	B
3	D
4	B
5	A
6	C
7	B
8	B
9	D
10	D
11	B
12	C
13	B
14	C
15	A

KAYNAKÇA

- SHEPHERD, George, **Microsoft ASP.NET 4.0 Step by Step**, Microsoft Press, Washington, 2010.
- SHARP, John, **Microsoft C# 2008 Step by Step**, Microsoft Press, Washington, 2008.