
WEB TABANLI PROGRAMLAMA

BÖLÜM 9

OLAYLAR, PENCERE VE FORMLAR

JAVASCRIPT'TE OLAY YÖNETİMİ,
PENCERE NESNELERİ, FORM İŞLEMLERİ

Prof. Dr. Turgay Tugay Bilgin

turgay.bilgin@btu.edu.tr



GENEL BAKIŞ...

9.1) Olay Kavramı

9.1.1) onclick Olayı

9.1.2) onsubmit Olayı

9.1.3) onmouseover ve onmouseout Olayları

9.1.4) Javascript Olayları Tablosu

9.2) JavaScript Pencere Nesneleri

9.2.1) window.alert() Penceresi

9.2.2) window.confirm() Penceresi

9.2.3) window.prompt() Penceresi

9.2.4) window.print() Penceresi

9.2.5) window.location() Metodu

9.3) JavaScript ile HTML Form Elemanlarından Bilgi Alma

9.3.1) DOM Seviye 1 Yöntemiyle Form Alanlarına Erişim

9.3.2) DOM Seviye 2 Yöntemiyle Form Alanlarına Erişim

9.3.3) Javascript ile Form Denetleme

9.4) Özet

-
- Programlama dillerinin gelişmeye başladığı yıllarda, program kodları, fonksiyonlar, nesneler gibi bileşenler programcı tarafından yazılır ve bunların ne zaman çağırılacağını yine programcı tarafından kod yazarak belirleniyordu. Gelişen teknoloji ile birlikte programlama teknolojileri ihtiyaçlar ve kullanıcı talepleri doğrultusunda değişmeye başladı. Özellikle görsel ara birimli işletim sistemlerinin ortaya çıkmasıyla birlikte kullanıcılar, fare tıklamaları, sürükleyip bırakmalar, liste öğelerini değiştirme, bir resme çift tıklama gibi etkileşimler sonucu olarak programların “bir şeyler” yapmalarını beklemeye başladılar. İşte tüm bu etkileşimlerin her birine olay adı verilir.
 - Olaylar ile ilgili kullanıcıya verilen geribildirimler çoğunlukla pencereler aracılığıyla verilir. Bu bölümde pencere nesnelerinin olaylar ile birlikte kullanımından bahsedilecektir.
 - HTML ünitesinde bahsedilen form bileşenleri ile etkileşimler de olaydır ve bunların kullanımı da bu bölümde açıklanmaktadır.

(9.1) OLAY KAVRAMI

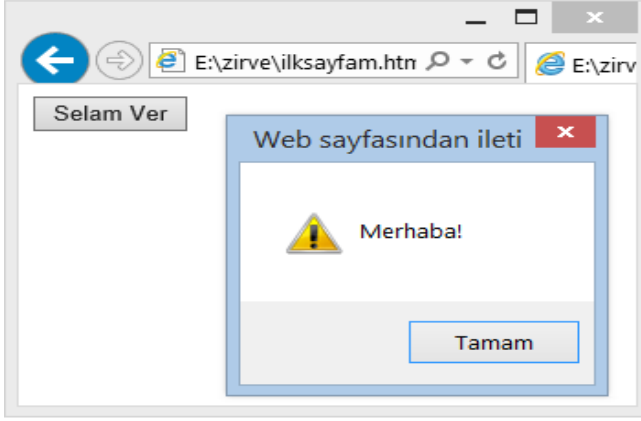
- İşletim sistemi, girdi araçları ya da diğer çevresel birimlerce oluşturulan etkiler ve aktivitelerin her biri olaydır. Örneğin kullanıcının fareyle ekrandaki bir butonu ya da bağlantıyı (linki) tıklaması bir olay meydana getirir.
- JavaScript dili de birçok modern programlama dilinde olduğu gibi “olay tabanlı” bir programlama dilidir. Yazılım geliştiriciler buton tıklamalarında pencerenin kapanması, mesaj görüntülenmesi, form girdilerinin kontrol edilmesi gibi işlemleri JavaScript olay denetçileri aracılığıyla kontrol edebilirler. İlerleyen bölümlerde JavaScript dilinde bulunan olaylardan en sık kullanılan bazıları açıklanmıştır.

-
- **9.1.1 onclick Olayı :** “onclick” olayı buton gibi bazı form elamanları veya HTML öğeleri tıklandığında oluşur. Hangi HTML bileşeni tıklanınca olay oluşacaksa ilgili elemanın “onclick” olayına, çalışması istenen JavaScript fonksiyonu adı yazılır.

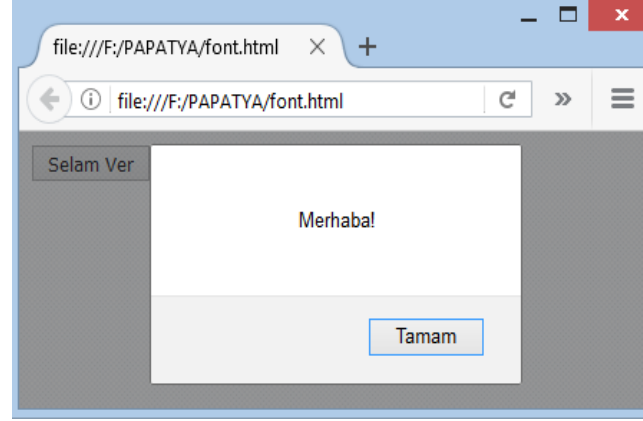
```
<html><head>
<script type="text/javascript">
  function Selamla()
  {
    alert("Merhaba!");
  }
</script>
</head><body>
<form>
<input type="button" onclick="Selamla()" value="Selam Ver">
</form>
</body></html>
```

Kod 9.1. onclick olayı örneği

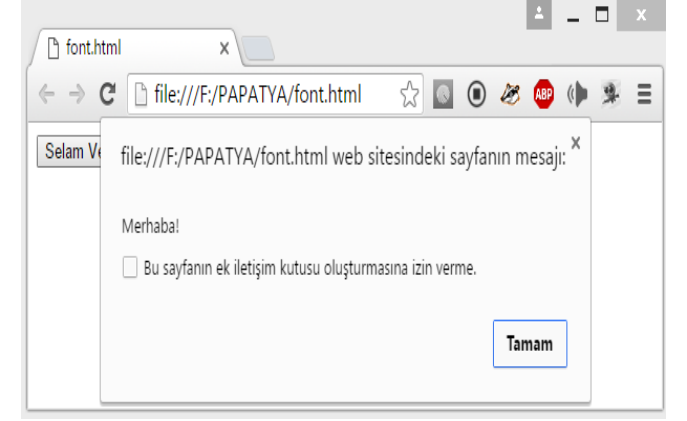
- Yukarıdaki kod parçası çalıştırıldığında “Selam ver” butonu tıklanınca butonun onclick niteliğinde tanımlı olan Selamla() fonksiyonu çağırılmakta ve sonuçta Şekil 9.1’deki gibi bir pencere içerisinde “Merhaba” yazısı görünmektedir.



(a) MS Internet Explorer



(b) Mozilla Firefox



(c) Google Chrome

Şekil 9.1. OnClick olayı örneğinin 3 farklı web tarayıcıda elde edilen görüntüsü

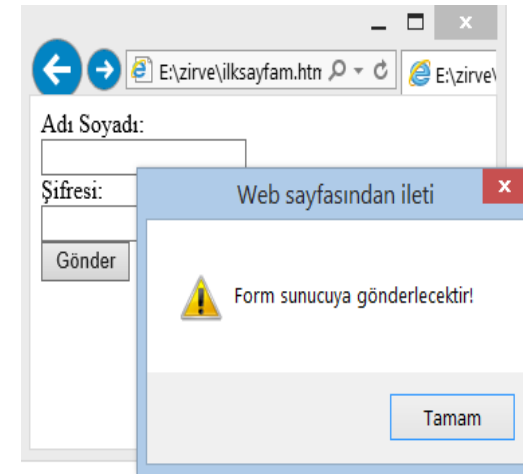
Hatırlatma ~ Javascript ile oluşturulan pencerelerin görünümü ve yerleşimleri farklı web tarayıcılarda farklı olabilir. Şekil 9.1’de 3 farklı tarayıcıda aynı kodun çalıştırılması ile elde edilen sonuç görülmektedir. Hepsinde ortak olan ekranda bir pencere görünmesi ve bu pencerede 1 adet “Tamam” butonu bulunmasıdır.

9.1.2. onsubmit Olayı : JavaScript'te diğer bir önemli olay ise *onsubmit* olayıdır. Bu olay bir HTML form elemanı sunucuya gönderilirken submit (gönder) butonuna basıldığında oluşur. Aşağıda buna ait bir örnek verilmiştir. Örnekte formu gönder butonuna tıklanınca `kontrolEt()` fonksiyonu çalıştırılıyor. `kontrolEt()` fonksiyonu **true** döndürürse form sunucuya postalanır, **false** döndürürse sunucuya postalama işlemi gerçekleştirilmez. Burada fonksiyon içinde herhangi bir koşul kontrolü yaptırmadığımız için ekranda “*Form sunucuya gönderilecektir!*” mesajı çıkacak ve `return true` ifadesinden dolayı form sunucudaki `kaydet.php` dosyasına gönderilmeye çalışılacaktır.

```
<html>
<head>
<script type="text/javascript">
function kontrolEt() {
    alert("Form sunucuya gönderilecektir!");
    return true;
}
</script>
</head>
<body>
<form name="giris" action="kaydet.php" method="POST"
onsubmit="return kontrolEt()">
Adı Soyadı:<br/><input type="text" name="adsoyad"><br/>
Şifresi:<br/><input type="password" name="sifre"><br/>
<input type="submit" value="Gönder" />
</form></body>
</html>
```

Kod 9.2. onSubmit olayı örneği

- Yandaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda Şekil 9.2'deki görüntü elde edilir.



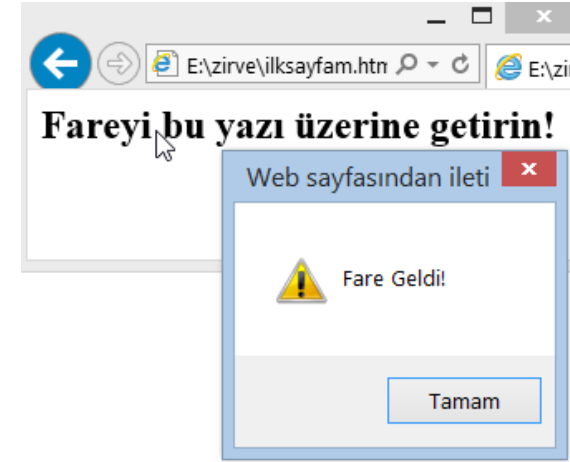
Şekil 9.2. JavaScript'te onsubmit örneği

9.1.3. *onmouseover* ve *onmouseout* Olayları : Bu iki olay farenin hareketi ile tetiklenir. *onmouseover* olayı fare bizim belirlediğimiz bir HTML elemanı üzerine geldiğinde tetiklenir. Burada önemli olan şudur: fare bir elemanın üzerine geldiğinde hiçbir tıklama işlemi oluşmasa da bu olay gerçekleşir. *onmouseout* olayı ise fare aynı HTML elemanı üzerinden ayrılması anında tetiklenir. Bu iki olay kullanılarak fare ile kontrol edilebilen çeşitli animasyonlar oluşturulabilir. Bu iki olayın kullanımını gösteren küçük bir örnek aşağıda verilmiştir.

```
<html>
<head>
<script type="text/javascript">
function gel () {
    alert("Fare Geldi!");
}
function git () {
    alert("Fare Gitti!");
}
</script>
</head>
<body>
<div onmouseover="gel()" onmouseout="git()">
<h2> Fareyi bu yazı üzerine getirin! </h2>
</div>
</body>
</html>
```

Kod 9.3. *onmouseover* ve *onmouseout* olayı örneği

- Yukarıdaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda Şekil 9.3'deki görüntü elde edilir.



Şekil 9.3. JavaScript'te *onmouseover* ve *onmouseout* örneği

9.1.4 JavaScript Olayları Tablosu :

HTML 4 ve yukarısı sürümlerin desteklediği JavaScript olaylarının tümü aşağıdaki tabloda kısa açıklamaları ile birlikte verilmiştir. Bunların herbirinin kullanımı yukarıda örneği verilen onclick, onsubmit, onmouseover ve onmouseout olaylarına benzer şekildedir. Detaylı bilgi için olayın adını <http://www.w3schools.com> sitesinde aratarak çeşitli örneklere ulaşabilirsiniz.

Olay	Tanımı
onchange	Metin kutusu gibi bir HTML elemanı içeriği değiştiğinde oluşur.
onsubmit	Form gönderildiğinde oluşur.
onreset	Form temizlendiğinde oluşur.
onselect	Bir HTML elemanı seçildiğinde oluşur.
onfocus	Bir HTML form elemanına odaklanıldığında (fare ile veya tab tuşu ile elemana bilgi girilmeye başlanacağı zaman) oluşur.
onblur	Bir HTML form elemanından başka bir form elemanına geçileceği zaman yani odaklanma biterken oluşur.
onkeydown	Bir HTML elemanı aktif durumdayken klavyeden bir tuşa basılırken oluşur.
onkeypress	Bir HTML elemanı aktif durumdayken klavyede bir tuşa basılı durduğu sürece oluşur.
onkeyup	Bir HTML elemanı aktif durumdayken klavyeden bir tuş bırakılırken oluşur.
onclick	Bir HTML elemanına fare ile tıklandığında oluşur.
onload	Bir HTML sayfası yüklenirken oluşur.
onmousedown	Fare tuşuna basılırken oluşur.
onmousemove	Fare bir HTML elemanı üzerinde hareket ederken oluşur.
onmouseout	Fare bir HTML elemanı üzerinden ayrılırken oluşur.
onmouseover	Fare bir HTML elemanı üzerine geldiğinde oluşur.
onmouseup	Fare tuşu bırakılırken oluşur.

(9.2) JAVASCRIPT PENCERE NESNELERİ

- JavaScript dilinde kullanıcı ile etkileşimde bulunmak için hazır nesneler bulunmaktadır. Bunlardan en sık kullanılanı “`window`” nesnesidir. “`window`” nesnesi sayesinde `alert` gibi mesaj pencereleri, onay pencereleri, bilgi giriş pencereleri ve yazdırma pencereleri oluşturulabilir. Aşağıda `window` nesnesinin içerdiği metotlar örneklerle verilmiştir.

(9.2.1) window.alert() PENCERESİ

- Önceki konularda `alert()` kullanımına ait örnekler verilmişti. `alert()` aslında `window` nesnesine ait bir metottur ve `window.alert()` şeklinde kullanılır. Fakat çoğunlukla nesne adı olan `window` ifadesi yazılmasa da yazılmış gibi çalışmaktadır.
- Yandaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda “Selam ver” butonu tıklanınca butonun onclick niteliğinde tanımlı olan `Selamla()` fonksiyonu çağırılacak ve sonuçta aşağıdaki gibi bir pencere içerisinde “Merhaba” yazısı görünecektir. Örnekteki fonksiyonda `window.alert("Merhaba!")` yerine `alert("Merhaba!")` yazılsa da aynı çıktı elde edilir.

Kod 9.4. `window.alert()` örneği

```
<html><head>
<script type="text/javascript">
    function Selamla()
    {
        window.alert("Merhaba!");
    }
</script>
</head><body>
<form>
<input type="button"
onclick="Selamla()" value="Selam Ver">
</form>
</body></html>
```

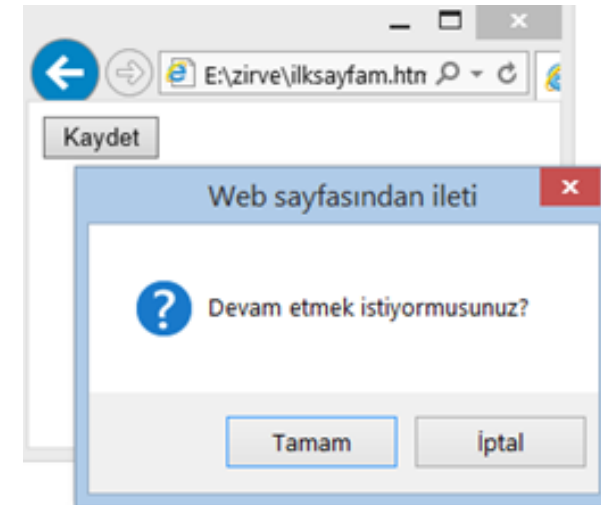
(9.2.2) `window.confirm()` PENCERESİ

- Bu pencere kullanıcıya “evet” veya “iptal” cevabı verebileceği bir soru sorabilmemize olanak sağlar. `window.confirm()` veya sadece `confirm()` ile oluşturulan onay penceresinde **OK** (tamam) ve **Cancel** (iptal) butonları bulunur. Eğer kullanıcı OK tuşuna tıklarsa `confirm()` metodu `true` değerini, **Cancel** tuşuna tıklarsa `confirm()` metodu `false` değerini döndürür.

```
<html><head>
<script type="text/javascript">
function onayla(){
    var cevap = confirm("Devam etmek
istiyormusunuz?");
    if( cevap == true ){
        alert("Kullanıcı devam etmek istiyor!");
        return true;
    }else{
        alert("Kullanıcı devam etmek istemiyor!");
        return false;
    }
}
</script>
</head><body>
<form>
<input type="button" value="Kaydet"
onclick="onayla();" />
</form></body>
</html>
```

Kod 9.5. window. confirm () örneği

- Yukarıdaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda Şekil 9.4'deki görüntü elde edilir.



Şekil 9.4. Javascript'te confirm örneği

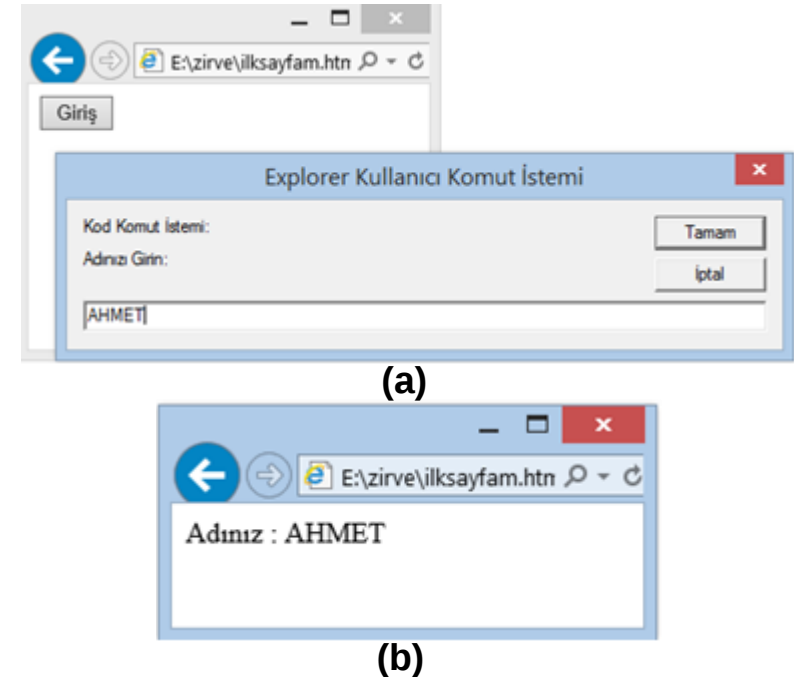
(9.2.3) window.prompt() PENCERESİ

- Web sayfalarında kullanıcıdan bilgi almak için bir metin kutusu yerine açılır bir pencere kullanabilirsiniz. Bu açılan pencereyi oluşturmayı sağlayan metot `window.prompt()` veya kısaca `prompt()` olarak bilinir. Bu iletişim kutusunda kullanıcıya vereceğiniz mesajı görüntüleyen bir etiket ve bir adet metin kutusu bulunur. Ayrıca **OK** (tamam) ve **Cancel** (iptal) butonları bulunur. Eğer kullanıcı metin kutusuna bir ifade girip **OK** (tamam) butonuna tıklarsa `prompt()` metodu girilen ifadeyi döndürür, eğer kullanıcı **Cancel** butonunu tıklarsa `prompt()` metodu *null* değeri döndürür.

Kod 9.6. window. prompt () örneği

```
<html>
<head>
<script type="text/javascript">
function degerAl () {
    var gelen = prompt("Adınızı Girin:",
    "buraya yazın..");
    document.write("Adınız : " + gelen );
}
</script>
</head>
<body>
<form>
<input type="button" value="Giriş"
onclick="degerAl();" />
</form>
</body>
</html>
```

- Yandaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda Şekil 9.5'teki görüntü elde edilir.



Şekil 9.5. *prompt* örneği: (a) Tamam tıklamadan önce, (b) “Ahmet” girip Tamam tıkladıktan sonra

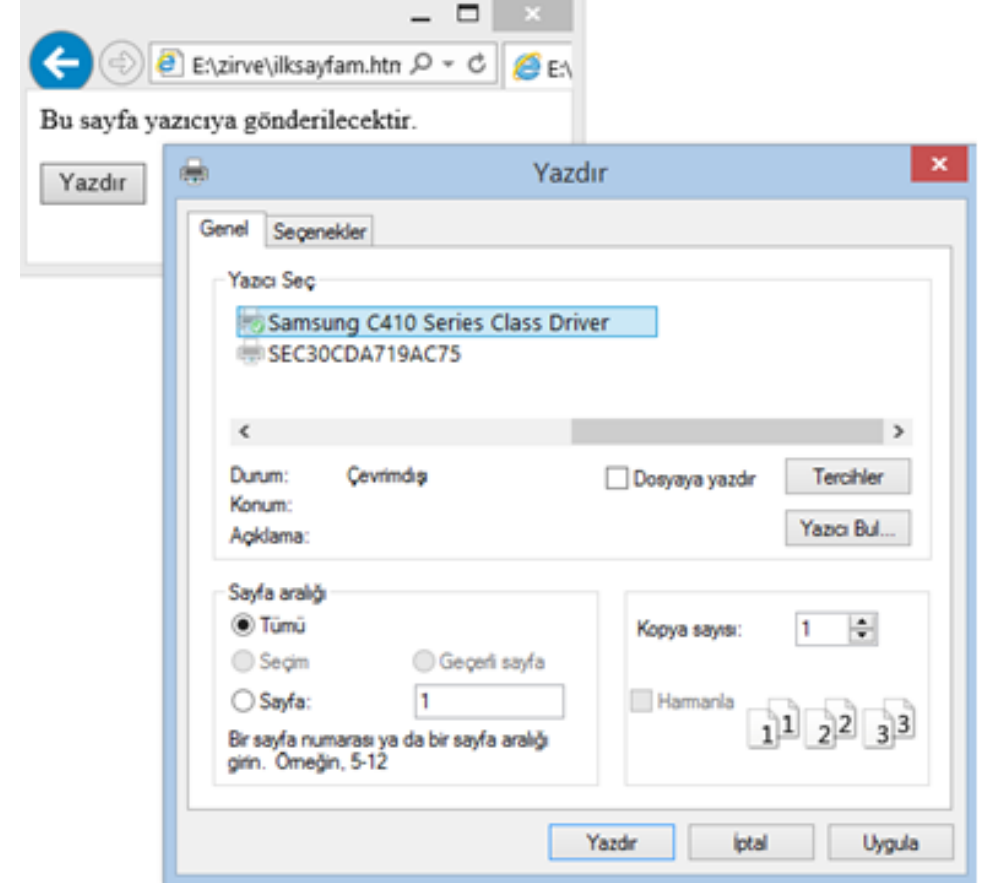
(9.2.4) window.print() PENCERESİ

- JavaScript window nesnesinin print metodunu kullanarak geçerli sayfayı yazdırmak için yazdırma penceresi görüntülenir. `window.print()` çalıştırıldığında geçerli web sayfasını yazdırmak işletim sistemine ait “yazdırma” penceresi görüntülenir. Bu pencerede bilgisayarınızda kurulu olan yazıcıyı seçerek geçerli web sayfasını yazdırabilirsiniz. Bu metodu doğrudan onclick olayı ile aşağıdaki gibi kullanabilirsiniz:

Kod 9.7.window. print () örneği

```
<html>
<head>
</head>
<body>
<p>Bu sayfa yazıcıya gönderilecektir.</p>
<form>
<input type="button" value="Yazdır"
onclick="window.print()" />
</form>
</body>
</html>
```

- Yukarıdaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda Şekil 9.6'teki görüntü elde edilir.

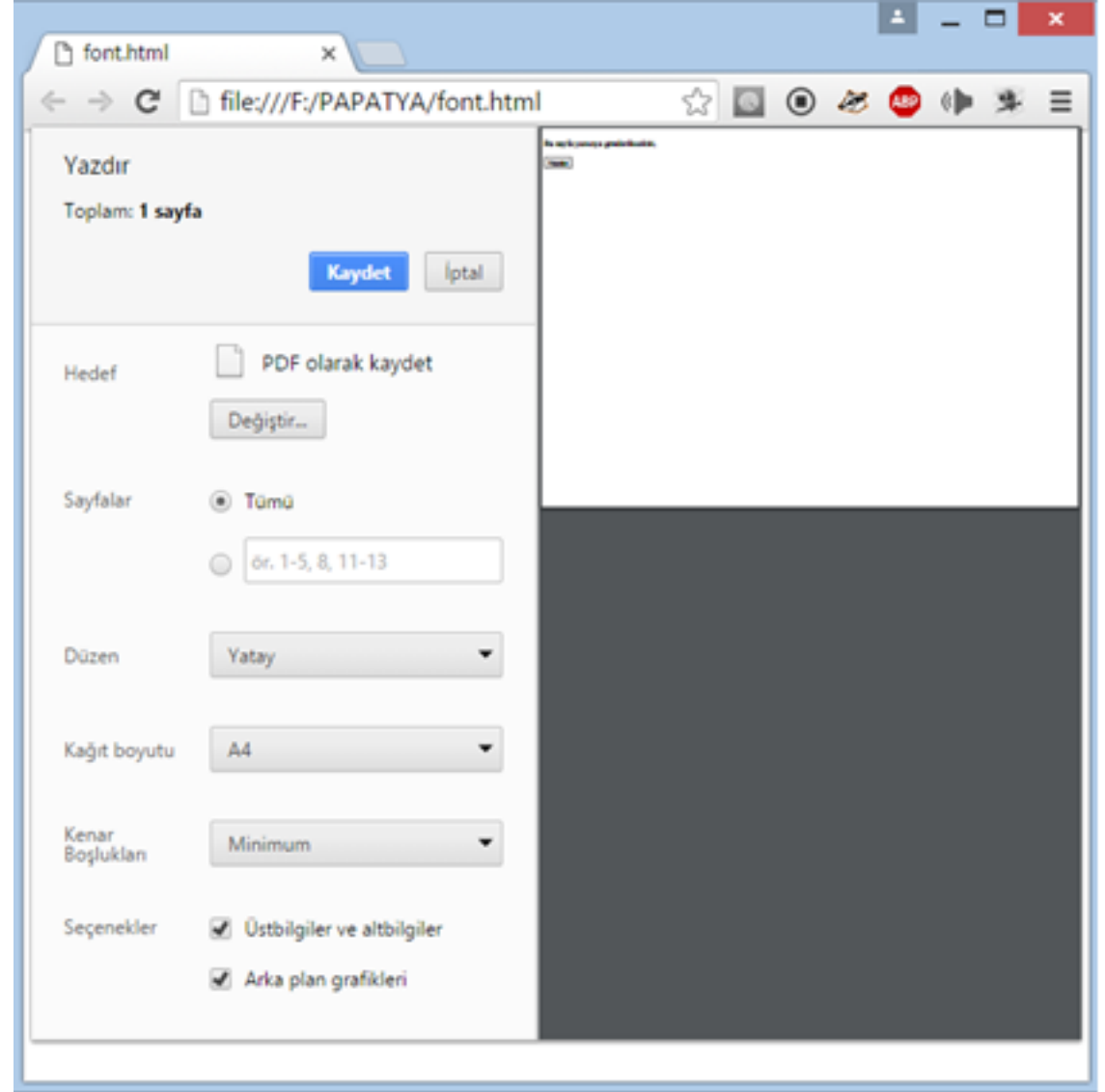


Şekil 9.6. Javascript'te *window.print* örneği

Hatırlatma *↪ Web tarayıcılardaki yazdırma pencereleri farkının sebebi*

Microsoft Internet Explorer (Microsoft Edge dahil) ve Mozilla Firefox web tarayıcıları işletim sisteminin sağladığı yazdırma penceresini kullanarak belge yazdırma işlemlerini gerçekleştirirler. Bu sebeple Mozilla Firefox'un farklı işletim sistemlerinde çalışan sürümlerinin oluşturduğu yazdırma pencereleri Şekil 9.6'daki pencereden farklı olabilir. Google Chrome tarayıcısı ise kendi özel yazdırma penceresini kullandığı için kurulduğu tüm işletim sistemlerinde aynı yazdırma penceresi görüntülenmektedir.

-
- Yazdırma örneği Google Chrome tarayıcısında çalıştırıldığında Şekil 9.7'deki yazdırma penceresi görüntülenir.



Bilgilendirme *↪ Bir yazıcı yok ise ne olacak?*

İşletim sisteminize bağlı bir yazıcı yok ise, yazdırma penceresinde yazıcı seçme bölümü boş olacağından yazdırma işlemi yapılamayacaktır. Bu durumda bazı web tarayıcıları PDF dosyaya yazdırma seçeneği sunmaktadırlar. Belge PDF dosyaya yazdırılarak daha sonra yazdırılmak üzere saklanabilir.

(9.2.5) `window.location()` METODU

- `window.location()` metodu sayesinde kullanıcı istenilen bir URL'ye yönlendirilebilir. Yönlendirme işleminin nasıl yapıldığını görmek için Kod 9.8'i inceleyiniz. HTML dili aracılığıyla da yönlendirme yapılabildiğini daha önceki konularda görmüştük. Fakat Javascript'de yönlendirme işlemi belirli bir koşula bağlı olarak da yapılabilir. Bunun için Kod 9.9'u inceleyiniz.

Kod 9.8. window.location() örneği

```
<html>
<head>
<script type="text/javascript">
function Yonlendir() {
    window.location="http://www.google.com.tr";
}
</script>
</head>
<body>
<form>
<input type="button" value="Yönlendir"
onclick="Yonlendir();" />
</form>
</body>
</html>
```

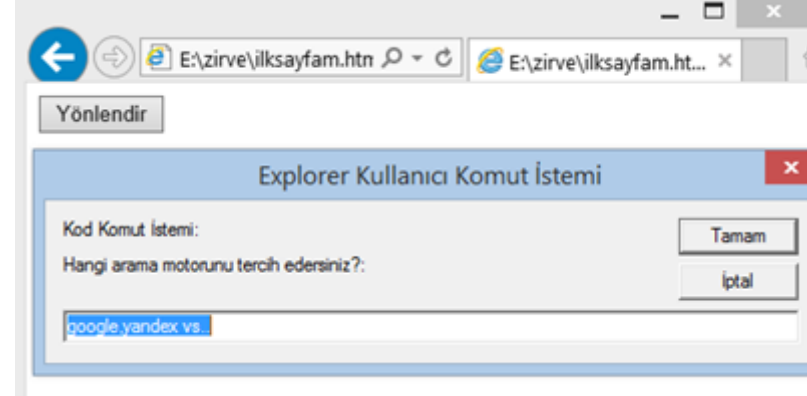
- Aşağıda verilen Kod 9.9'da kullanıcılar **Yönlendir** butonuna tıklayınca sayfa `http://www.google.com.tr` adresine yönlenecektir.

```
<html>
<head>
<script type="text/javascript">
function Yonlendir() {
    var gelen = prompt("Hangi arama motorunu tercih edersiniz?:", "google,yandex vs..");

    if( gelen == "google" ){
        window.location="http://www.google.com";
    }
    else if ( gelen =="yandex"){
        window.location="http://www.yandex.com";
    }
    else{
        alert("Hatalı giriş!");
    }
}
</script>
</head>
<body>
<form>
<input type="button" value="Yönlendir" onclick="Yonlendir();" />
</form>
</body>
</html>
```

Kod 9.9. window.location() ile koşullu yönlendirme

-
- Yukarıdaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda Şekil 9.8’teki görüntü elde edilir.
 - “*Hangi arama motorunu tercih edersiniz?*” sorusuna google cevabı verilmiş ise sayfa <http://www.google.com> sitesine, yandex cevabı verilmiş ise <http://www.yandex.com> sitesine yönlenecektir. Bunların dışında bir giriş yapıldığında “Hatalı Giriş” mesajı görüntülenecek ve yönlendirme yapılmayacaktır.



Şekil 9.8. Javascript’te *window.location* örneği

(9.3) JAVASCRIPT İLE HTML FORM ELEMANLARINDAN BİLGİ ALMA

- HTML formları ile kullanıcıdan bilgi alınması sırasında kullanıcılar hatalı bilgi girişi yapabilir. Bu hatalar iki farklı aşamada denetlenebilir:
 - 1) Form bilgileri web tarayıcı tarafından henüz sunucuya gönderilmemişken (istemci tarafında)
 - 2) Form bilgileri web sunucuya gönderildiğinde (sunucu tarafında)
- Bu iki yöntemden birincisi yani istemci tarafında kontrol çok daha hızlı ve avantajlıdır. Çünkü denetleme işlemi henüz bilgiler sunucuya gitmemişken yapılır. İkinci yöntemde ise bilgilerin sunucuya ulaşması hem bir miktar zaman alır hemde gereksiz internet trafiği ve sunucunun gereksiz yere meşgul edilmesine sebep olur.
- Birinci aşamada bilgileri denetlemek için Javascript, ikinci aşamada ise sunucu tabanlı denetleme yapılacağından ilerleyen bölümlerde göreceğimiz PHP dili kullanılır. Bu bölümde Javascript ile form bilgilerini denetlemeyi göreceğiz.

-
- JavaScript gibi nesneye yönelik programlama dillerinin, HTML gibi nesne kavramı olmayan diller ile bilgi alış verişi yapabilmesi için DOM (Belge Nesne Modeli - Document Object Model) adı verilen nesne yapılandırma standardı kullanılır. DOM, HTML ile programlama dilleri arasında bir standart oluşturarak bu dillerin HTML den bilgi alışverişini sağlar. DOM, Nesneler ve özelliklerden oluşur.
 - JavaScript ile HTML form öğelerine erişim, DOM sürümüne göre iki farklı türde gerçekleştirilebilir:
 - **DOM seviye 1** olarak adlandırılan eski yöntemde HTML elemanın `name` niteliği kullanılır. Bu yöntem eski olmasına rağmen web tarayıcıları tarafından desteklenmeye devam etmektedir.
 - **DOM seviye 2** olarak adlandırılan yeni yöntemde ise HTML elemanın `id` niteliği kullanılır. Bu yöntem 2003 yılından itibaren üretilen web tarayıcılar tarafından desteklenmektedir.

9.3.1 DOM Seviye 1 Yöntemiyle Form Alanlarına Erişim : DOM seviye 1 ile Form alanlarına erişebilmek için öncelikle form etiketinin `name` özelliğine mutlaka bir isim verilmelidir. Ayrıca tüm HTML form elemanlarına da `name` özelliği kullanılarak isim verilmelidir. İsim verilmeyen form elemanlarına JavaScript ile erişmek mümkün değildir.

- JavaScript ile metin kutusu, açılan kutu, parola kutusu veya çok satırlı metin kutusunun içindeki değeri okuyabilmek için

```
document.FormunAdı.ElemanınAdı.value
```

ifadesi kullanılır. Eğer bir form elemanına değer verilecekse de aynı şekilde `value` özelliğine atama yapılır. Eğer erişilecek eleman checkbox veya radiobutton ise

```
document.FormunAdı.ElemanınAdı.checked
```

ifadesi ile seçili olup olmadığı kontrol ettirilir. Genellikle aşağıdaki örnekteki gibi seçili olma durumuna göre bir if yapısı içerisinde işlem gerçekleştirilir.

```

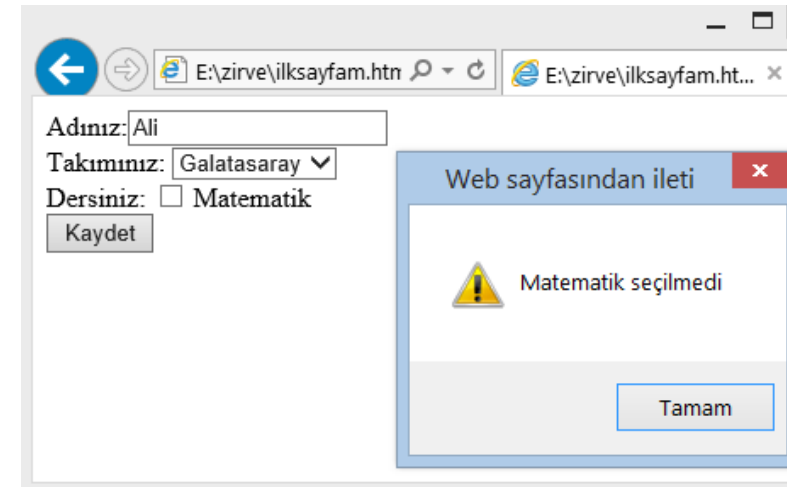
<html>
<head>
<script type="text/javascript">
function Denetle()
{
    alert("Adınız:" + document.KayitFormu.isim.value);
    alert("Takım No:" + document.KayitFormu.takim.value);

    if (document.KayitFormu.matematik.checked) {
        alert("Matematik seçildi");
    } else {
        alert("Matematik seçilmedi");
    }
    return true;
}
</script>
</head>
<body>
<form name="KayitFormu" onsubmit="return Denetle();">
Adınız:<input type="text" name="isim" />
<br />
Takımınız: <select name="takim">
    <option value="0"> Seçiniz</option>
    <option value="1">Fenerbahçe</option>
    <option value="2">Galatasaray</option>
    <option value="3">Beşiktaş</option>
</select><br />
Dersiniz: <input type="checkbox" name="matematik">
Matematik
<br /><input type="submit" value="Kaydet" />
</form>
</body>
</html>

```

Kod 9.10. DOM seviye 1 yöntemiyle form alanlarına erişim

- Yandaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda Şekil 9.9'daki görüntü elde edilir. Bu örnekte form içindeki **isim** alanı ve **takım** alanına ait "value" değerleri ekrana yazdırılıyor ve **dersiniz** alanında Matematik seçiliyse ekranda "Matematik seçildi" bilgisi görüntüleniyor. Matematik seçili değil ise ekranda "Matematik seçilmedi" bilgisi görüntülenmektedir.



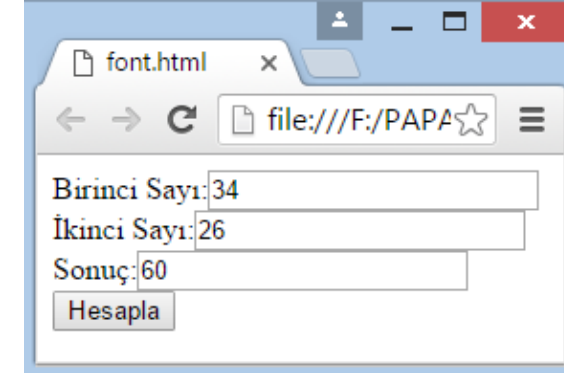
Şekil 9.9. JavaScript ile form alanlarına erişim örneği

- Aşağıda verilen Kod 9.11’de ise basit bir hesap makinası oluşturulmuştur. HTML input elemanları ile kullanıcıdan alınan 2 sayı toplanarak üçüncü input elemanının içine yazdırılmaktadır.

```
<html>
<head>
<script type="text/javascript">
function hesapla(){
    var a=document.hesapmak.sayi1.value;
    var b=document.hesapmak.sayi2.value;
    var toplam= parseInt(a)+parseInt(b);
    document.hesapmak.sonuc.value=toplam;
}
</script>
</head>
<body>
    <form name="hesapmak" >
    Birinci Sayı:<input type="text" name="sayi1" /><br />
    İkinci Sayı:<input type="text" name="sayi2" /><br />
    Sonuç:<input type="text" name="sonuc" /><br />
    <input type="button" value="Hesapla" onclick="hesapla();" />
    </form>
</body>
</html>
```

Kod 9.11. DOM seviye 1 yöntemiyle hesap makinası

- Yandaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda Şekil 9.10’daki görüntü elde edilir.



Şekil 9.10. JavaScript ile basit hesap makinası örneği

Bilgilendirme *HTML form elemanlarından alınan bilgilerin veri tipi nedir?*

HTML form elemanlarından alınan sayısal değerler daima metin biçimindedir. Metin biçiminde depolanan sayılar ile matematik işlemler yapılamayacağından bunların sayısal tiplere dönüştürülmesi gerekir. Javascript dilinde metin biçiminde depolanan sayıları sayısal tipe çevirmek için parseInt() metodu kullanılır. Bu metod, parametre olarak aldığı metin biçimli sayıyı sayısal biçime dönüştürür.

9.3.2 DOM Seviye 2 Yöntemiyle Form Alanlarına Erişim : DOM seviye 2 ile Form alanlarına erişebilmek için form etiketinin `id` özelliğine mutlaka bir değer verilmelidir. Ayrıca tüm HTML form elemanlarına da `id` özelliği kullanılarak isim verilmelidir. “`id`” verilmeyen form elemanlarına JavaScript ile erişmek mümkün değildir.

- JavaScript ile metin kutusu, açılan kutu, parola kutusu veya çok satırlı metin kutusunun içindeki değeri okuyabilmek için

```
document.getElementById("...").value
```

ifadesi ile erişilir. Burada “...” yerine erişilecek elemanın `id` değeri yazılır. Eğer bir form elemanına değer verilecekse de aynı şekilde `value` özelliğine atama yapılır. Eğer erişilecek eleman checkbox veya radiobutton ise

```
document.getElementById("...").checked
```

ifadesi ile seçili olup olmadığı kontrol ettirilir. Genellikle Örnek 1’deki gibi seçili olma durumuna göre bir if yapısı içerisinde işlem gerçekleştirilir.

Kod 9.12. DOM seviye 2 yöntemiyle form alanlarına erişim

```
<html>
<head>
<script type="text/javascript">
function Denetle()
{
    alert("Adınız:" + document.getElementById("isim").value);
    alert("Takım No:" + document.getElementById("takim").value);

    if (document.getElementById("matematik").checked){
        alert("Matematik seçildi");
    } else{
        alert("Matematik seçilmedi");
    }
    return true;
}
</script>
</head>
<body>
<form name="KayitFormu" onsubmit="return Denetle();">
Adınız:<input type="text" id="isim" />
<br />
Takımınız: <select id="takim">
    <option value="0"> Seçiniz</option>
    <option value="1">Fenerbahçe</option>
    <option value="2">Galatasaray</option>
    <option value="3">Beşiktaş</option>
</select><br />
Dersiniz: <input type="checkbox" id="matematik">
Matematik
<br /><input type="submit" value="Kaydet" />
</form>
</body>
</html>
```

- Yukarıdaki kodu **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda elde edilecek çıktı, Şekil 9.9'da görülen çıktı ile birebir aynı olacaktır.

- Aşağıda verilen Kod 9.13'de DOM seviye 2 yöntemiyle basit bir hesap makinası oluşturulmuştur. HTML input elemanları ile kullanıcıdan alınan 2 sayı toplanarak üçüncü input elemanının içine yazdırılmaktadır.

```
<html>
<head>
<script type="text/javascript">
function hesapla() {
    var a=document.getElementById("sayi1").value;
    var b=document.getElementById("sayi2").value;
    var toplam= parseInt(a)+parseInt(b);
    document.getElementById("sonuc").value=toplam;
}
</script>
</head>
<body>
<form id="hesapmak" >
Birinci Sayı:<input type="text" id="sayi1" /><br />
İkinci Sayı:<input type="text" id="sayi2" /><br />
Sonuç:<input type="text" id="sonuc" /><br />
<input type="button" value="Hesapla" onclick="hesapla();" />
</form>
</body>
</html>
```

Kod 9.13. DOM seviye 2 yöntemiyle hesap makinası

- Kod 9.13'ün çıktısı, Şekil 9.10'da görülen çıktı ile birebir aynı olacaktır.

9.3.3 JavaScript ile Form Denetleme : JavaScript dilinin en yoğun kullanıldığı alanlardan biri istemci taraflı form denetlemedir. Kod 9.13’de verilen örnekte JavaScript ile form denetleme işlemi gerçekleştirilmiştir. Her bir form elemanının boş olup olmadığına bakılıyor, eğer boş ise `alert` ile bilgilendirme mesajı veriliyor ve `focus()` metodu aracılığıyla boş olan elemana farenin odaklanması sağlanıyor. Böylece kullanıcının boş olan alanı doldurması isteniyor. Bir form alanı doldurulmadığı sürece `return false` ifadesi sayesinde formun sunucuya gönderilmesi engelleniyor.

- Posta kodu alanı eğer boş ise veya girilen değer sayı değil ise veya girilen değerın uzunluğu 5’ten farklı ise uyarı görüntüleyecek ve formu göndermeyecektir. Ayrıca “ülke” seçim kutusunda “seçiniz” ifadesinin değeri “-1” olarak belirlendiğinden, eğer bu ifade “-1” olarak kalmış ise kullanıcının ülke seçmemiş olduğu anlaşılır. Bu durumda kullanıcı ülke seçmeye zorlamak için kullanıcıya “ülke seçiniz” mesajı verilmektedir.

Bilgilendirme  isNaN fonksiyonu nasıl kullanılır?

isNaN fonksiyonu, parametre olarak aldığı değer sayı ise false, sayı değil ise true döndüren bir kontrol fonksiyonudur.

```

<html>
<head>
<script type="text/javascript">
function Denetle()
{
    if( document.KayitFormu.adi.value == "" )
    {
        alert( "Lütfen adınızı girin!" );
        document.KayitFormu.adi.focus();
        return false;
    }
    if( document.KayitFormu.email.value == "" )
    {
        alert( "Lütfen e-posta adresi girin!" );
        document.KayitFormu.email.focus();
        return false;
    }
    if( document.KayitFormu.postakodu.value == "" ||
        isNaN( document.KayitFormu.postakodu.value ) ||
        document.KayitFormu.postakodu.value.length != 5
    )
    {
        alert( "Lütfen posta kodunu 5 haneli olarak girin!" );
        document.KayitFormu.postakodu.focus();
        return false;
    }
    if( document.KayitFormu.ulke.value == "-1" )
    {
        alert( "Lütfen ülke seçiniz!" );
        return false;
    }
    return true;
}
</script>
</head>

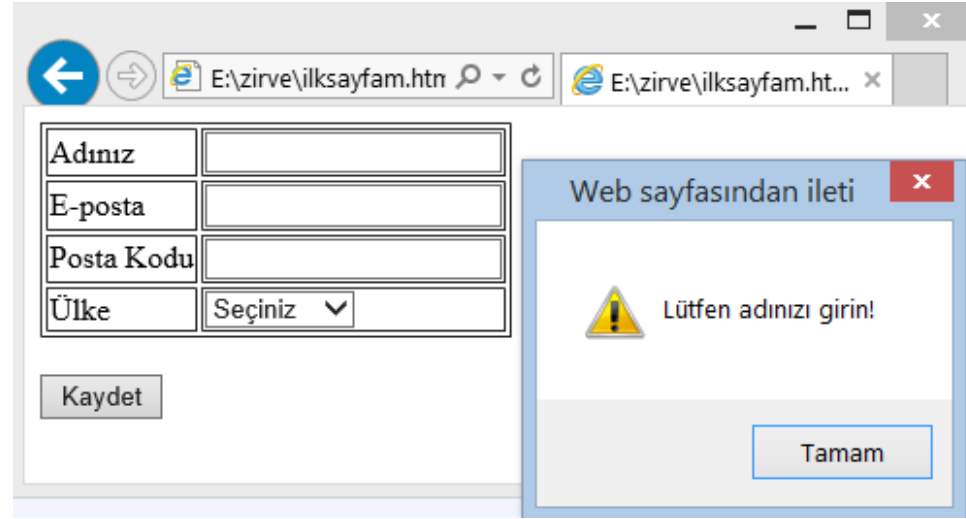
```

```

<body>
<form action="" name="KayitFormu"
onsubmit="return Denetle();">
<table border="1">
<tr>
<td>Adınız</td>
<td><input type="text" name="adi" /></td>
</tr>
<tr>
<td>E-posta</td>
<td><input type="text" name="email" /></td>
</tr>
<tr>
<td>Posta Kodu</td>
<td><input type="text" name="postakodu" /></td>
</tr>
<tr>
<td>Ülke</td>
<td>
<select name="ulke">
<option value="-1"
selected>Seçiniz</option>
<option value="1">Türkiye</option>
<option value="2">Almanya</option>
<option value="3">İtalya</option>
</select>
</td>
</tr>
</table>
<br /><input type="submit" value="Kaydet" />
</body>
</html>

```

- Yukarıdaki kodları **sayfam.html** adıyla kaydederek web tarayıcısında çalıştırdığınızda Şekil 9.11'deki görüntü elde edilir.



Şekil 9.11. JavaScript ile form denetleme örneği

- Böylece formdaki bütün bilgiler tam olarak doldurulduktan sonra sunucuya gönderilecektir. Sunucuda bu bilgilerin nasıl işleneceğini bir sonraki ünite olan PHP konusuna geldiğinizde öğreneceksiniz.

(9.4) ÖZET

- Girdi aygıtları ya da diğer çevresel birimlerce oluşturulan etkiler ve aktivitelerin her biri olaydır. JavaScript dili de birçok modern programlama dilinde olduğu gibi “olay tabanlı” bir programlama dilidir. “onclick” olayı buton gibi bazı form elamanları veya HTML öğeleri tıklandığında oluşur. JavaScript’te diğer bir önemli olay ise onsubmit olayıdır. Bu olay bir HTML form elemanı sunucuya gönderilirken submit (gönder) butonuna basıldığında oluşur. Bu iki olay farenin hareketi ile tetiklenir. onmouseover olayı fare bizim belirlediğimiz bir HTML elemanı üzerine geldiğinde tetiklenir. onmouseout olayı ise fare aynı HTML elemanı üzerinden ayrılması anında tetiklenir. JavaScript dilinde kullanıcı ile etkileşimde bulunmak için hazır nesneler bulunmaktadır. Bunlardan en sık kullanılanı “window” nesnesidir. “window” nesnesi sayesinde alert gibi mesaj pencereleri, onay pencereleri, bilgi giriş pencereleri ve yazdırma pencereleri oluşturulabilir. JavaScript window nesnesinin print metodunu kullanarak geçerli sayfayı yazdırmak için yazdırma penceresi görüntülenir. window.location() metodu sayesinde kullanıcı istenilen bir URL’ye yönlendirilebilir. JavaScript ile Form alanlarına erişebilmek için öncelikle form etiketinin name özelliğine mutlaka bir isim verilmelidir. Ayrıca tüm HTML form elemanlarına da name özelliği kullanılarak isim verilmelidir. İsim verilmeyen form elemanlarına JavaScript ile erişmek mümkün değildir.