



Atatürk Üniversitesi
Açıköğretim Fakültesi

İnternet Programcılığı II



Bu kitabın, basım, yayım ve sa- hakları Atatürk Üniversitesi'ne ai- r. Bireysel ö- renme
yakla- imıyla hazırlanan bu kitabı bütün hakları saklıdır. Atatürk Üniversitesi'nin izni
almamaksızın kitabı tamamı veya bir kısmı mekanik, elektronik, fotokopi, manye- k kayıt veya
ba- ka- ekillerde ço- al- lamaz, basılamaz ve da- lamaz.

Copyright © 2020

The copyrights, publica- ons and sales rights of this book belong to Atatürk University. All rights reserved of this book prepared with an individual learning approach. No part of this book may be reproduced, printed, or distributed in any form or by any means, technical, electronic, photocopying, magne- c recording, or otherwise, without the permission of Atatürk University.



ATATÜRK ÜN VERS TES AÇIKÖ- RET M FAKÜLTES

İnternet Programalı İ II

ISBN: 978-605-7638-75-5

ATATÜRK ÜN VERS TES AÇIKÖ- RET M FAKÜLTES YAYINI

ERZURUM, 2020

➤ İÇİNDEKİLER

1. HTML'e giri <i>Doç. Dr. brahim Çetin</i>	<u>4</u>
2. HTML5 ile Gelen Yenilikler <i>Doç. Dr. brahim Çetin</i>	<u>27</u>
3. HTML5 ile Yeni Veri Giriş ve Form Elemanları <i>Doç. Dr. brahim Çetin</i>	<u>48</u>
4. HTML5 ile İleri Teknolojiler <i>Doç. Dr. brahim Çetin</i>	<u>67</u>
5. CSS ve Metin Stilleri <i>Doç. Dr. Ercan TOP</i>	<u>87</u>
6. CSS'te Renk Tanımı ve Tablo Düzenleme <i>Doç. Dr. Ercan TOP</i>	<u>114</u>
7. Liste ve Görsellerde CSS Stilleri <i>Doç. Dr. Ercan TOP</i>	<u>141</u>
8. CSS'te Kutu Modeli, Kalıtsallık ve Öncelikler <i>Doç. Dr. Ercan TOP</i>	<u>168</u>
9. Javascript ile Kodlamaya Giri <i>Dr. Ö. r. Üyesi HAL LERSOY</i>	<u>195</u>
10. Javascript Dili ve Komutları <i>Dr. Ö. r. Üyesi HAL LERSOY</i>	<u>216</u>
11. Javascript'te İleri Konular <i>Dr. Ö. r. Üyesi HAL LERSOY</i>	<u>244</u>
12. JQuery'e Giri <i>Ö. r. Gör. Rafet Orçun MADRAN</i>	<u>280</u>
13. JQuery Mobile ile Tasarım <i>Ö. r. Gör. Rafet Orçun MADRAN</i>	<u>309</u>
14. Bootstrap ile Duyarlı Web Tasarımı <i>Ö. r. Gör. Rafet Orçun MADRAN</i>	<u>339</u>

Editör

Prof. Dr. Erman Yükseltürk

HTML'E GİRİŞ

İÇİNDEKİLER

- HTML Nedir?
- HTML'in Temel Özellikleri
- HTML'in Temel Etiketleri



HEDEFLER

- Bu üniteyi çalıştıktan sonra;
 - HTML dilinin tarihçesini açıklayabilecek,
 - HTML etiketlerini yazmak ve çalıştmak için gerekli programları seçebilecek,
 - HTML ile yapılabilecek işlemleri planlayabilecek,
 - HTML ile sayfalar tasarlamayı öğrenebileceksiniz.

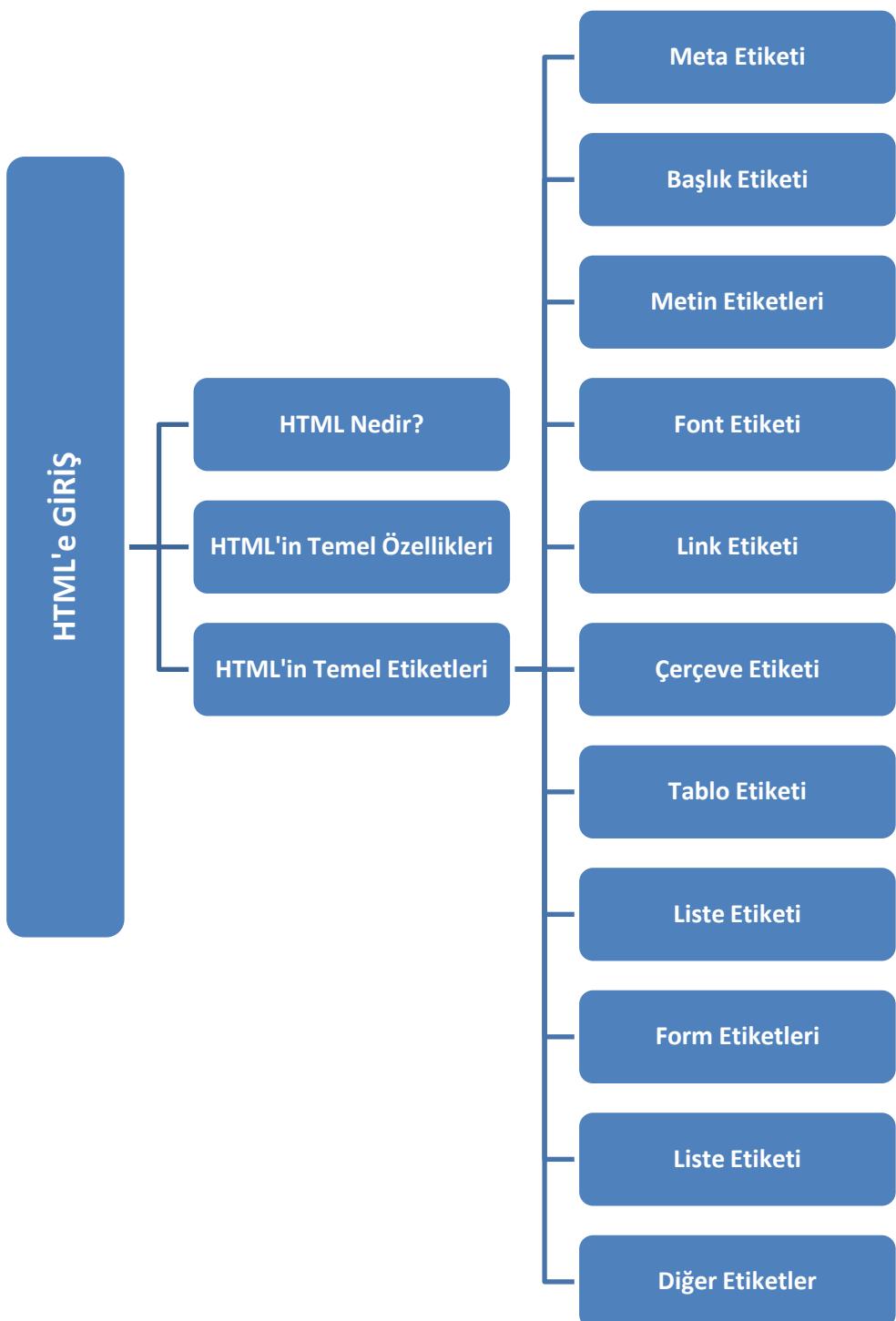


Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET
PROGRAMCILIĞI II
Doç. Dr. İbrahim
ÇETİN

ÜNİTE

1



GİRİŞ

Internet teknolojisi, ABD'de savunma sistemlerinde kullanılmak üzere ortaya çıkmış olsa da zamanla herkesin kullanımına açılmıştır. 1990'lı yıllarda WWW ve HTML teknolojilerinin gelişmesiyle birlikte internet hızla yayılmıştır. Şu an en ücra köşelere bile giren internet, artık milyarlarca insanın cep telefonlarında veya dizüstü bilgisayarlarında kolayca ulaşılabilir durumdadır. Internetin kullanımı yaygınlaşıırken internet tabanlı programlama dillerinde de hızlı bir gelişim meydana gelmiştir. Internet tabanlı programlamanın tarihine bakıldığından 25-30 yıllık bir geçmişine rağmen bu alanda çok çeşitli programlama dilleri geliştirilmiştir. Internet tabanlı programlamanın ilk yıllarından itibaren en bilinen dilinin HTML olduğu söylenebilir. Takip eden yıllarda CSS, Javascript gibi farklı teknolojiler de yaygınlaşmıştır. Bu kitapta HTML ile başlanacak, sonraki bölgümlerde CSS ve Javascript konuları gibi HTML ile beraber kullanılan diğer teknolojilerle devam edilecektir. Günümüz HTML dili tek başına kullanılmamakta, özellikle profesyonel tasarımların hemen hemen hepsinde CSS, Javascript ve JQuery gibi güncel teknolojilerle beraber kullanılarak çok daha etkileşimli sayfalar oluşturulmaktadır. Internet tabanlı programlama yapmak isteyenlerin bu teknolojileri öğrenmeleri beklenmektedir. Bu kitap, okuyucularına bu teknolojiler hakkında geniş bir kaynak imkânı sağlayacaktır. Özellikle hem kodların hem de web sayfası çıktılarını içeren örneklerinin olduğu, konuların sonlarında farklı bireysel aktivitelerle okuyucuları yönlendiren zengin bir içerik sunmaktadır. Bu bölümde HTML'e giriş yapılarak, HTML'in temel özellikleri anlatılacak ve en yaygın bilinen etiketler hakkında bilgiler verilecektir.

HTML NEDİR?



HTML'in açılımı: Hyper Text Markup Language. Türkçe meali ise: Zengin Metin İşaretleme Dili'dir.

HTML, Hyper Text Markup Language kelimelerinin kısaltılmasıından oluşan, web sayfalarını hazırlamak için kullanılan standart metin tabanlı işaretleme dilidir. Türkçe anlamı; Zengin Metin İşaretleme Dili'dir. HTML'in bir programlama dili olmadığı, daha çok betik bir dil olarak kabul edildiği söylenebilir. Başka bir ifadeyle HTML, Internet Explorer (IE), Chrome, Opera, Firefox, Safari gibi web tarayıcılarının okuyup anladığı bir dildir. Tarayıcılar web sitelerini HTML olarak sunarlar. Bunların dosya uzantıları genellikle ".html" veya ".htm" şeklinde tanımlanır. Web projeleri hangi yazılım dili ile geliştirilirse geliştirilsin, sonunda HTML olarak tarayıcıya yazılır ve kullanıcılar sayfaları HTML çıktı olarak görürler. Özetle; ASP, ASP.net, PHP, Java, Python gibi günümüzde kullanılan yazılım dilleri ile geliştirilen web projelerinin hepsinin ortak noktası HTML'dir.

HTML'in tarihçesi incelemişinde 1980'li yıllara kadar gidilmektedir. Aslında bunun internetin başlangıç yıllarına kadar gittiği söylenebilir. Örneğin, 1980 yılında Tim Berners-Lee, CERN'de çalışan araştırmacıların arasında bilgi ve dokümanların paylaşılabilmesi için bir çalışma başlatmıştır. ENQUIRE adının verildiği bu proje, HTML ile ilgili yapılan ilk çalışmalarдан biri olarak bilinmektedir. 1980'li yılların sonlarına gelindiğinde ise yine Tim Berners-Lee tarafından internet tabanlı sistemin kullanıldığı ve 1990'lı yıllarda ise World Wide Web (WWW) sisteminin kurulduğu görülmektedir. Aynı yıllarda HTML'in 18 temel öğesini tanımlayan



HTML ilk defa 1980'li yıllarda Tim Berners-Lee'nin önderliğini yaptığı grup tarafından ortaya atılmıştır.

"HTML Etiketleri" adlı bir doküman oluşturulmuştur. 1990'lı yılların ortalarında ise HTML Çalışma Grubu oluşturulmuş ve HTML 2.0'in çalışmalarına başlanmıştır. 2000'li yıllarda ise HTML'in 4.0 sürümü çıkarılmış ve artık HTML dili uluslararası bir standart hâline gelmiştir. 2010'lu yılların sonlarında ise HTML 5.0 sürümü üzerinde çalışmalar sonlandırılarak piyasaya sürülmeye hazır hâle getirilmiştir. Özette, HTML artık hayatımızın her aşamasında olan internetin temelini oluşturmuştur. Aşağıdaki tabloda yıllara göre sürümlerindeki gelişim verilmiştir.

Tablo 1.1. HTML'in tarihsel gelişimi

Sürüm	Tahmini Çıkış Yılları
HTML	1990'lı yılların başı
HTML 2.0	1990'lı yılların ortası
HTML 4.0	1990'lı yılların sonu
XHTML	2000'li yılların başı
HTML5	2010'lu yılların başı

HTML'İN TEMEL ÖZELLİKLERİ

HTML dili etiket yapısından oluşmaktadır. Genel olarak her satırı "*<etiket></etiket>*" şeklinde tanımlanmış kod satırları arasında işlemler yapılır. Başka bir ifadeyle işlemler, etiketlerin açılması "*<etiket>*" ve kapatılması "*</etiket>*" arasında gerçekleştirilir. Aşağıda HTML'in temel özellikleri maddeler hâlinde özetlenmiştir:

- HTML etiketleri 2 karakter ile sınırlanır: < ve >
- HTML etiketleri çift olarak kullanılır. Örn: <u> Bu metnin altı çizgiliidir. </u>
- Etiket çiftlerinden ilki başlama etiketi, ikincisi ise bitiş etiketi olarak adlandırılır.
- Başlama ve bitiş etiketleri arasında kalan her şeye element ya da contents, Türkçe olarak ise öge içeriği denir.
- HTML etiketleri büyük ya da küçük harfe duyarlı değildir, yani
 ve
 aynı görevi görür.

HTML ile bir web sayfası tasarlama için aslında hiçbir programa ihtiyaç yoktur. Bilgisayarlarınızda yüklü olan not defteri programını bile kullanarak HTML etiketlerini yazabilirsiniz. Ayrıca, görsel olarak zengin arayüze sahip kullanıcı dostu profesyonel editörlerle de web sayfaları tasarlabilir. Programlara örnek vermek gerekirse;

- Notepad++
- Sublime Text
- Microsoft Expression Web
- Adobe Dreamweaver
- Rapid PHP Editör
- CoffeeCup HTML Editor



HTML ile web sayfası hazırlamak için bilgisayarınızdaki not defteri programı yeterlidir.

Bu programların bazıları ücretli olabilmektedir. İlgili programların web sayfaları ziyaret edilerek daha detaylı bilgi alınabilir, istenirse deneme sürümleri indirip kullanılabilir. En kolay erişilebilen, ücretsiz ve yaygın olarak kullanılan Notepad++ programıdır. Profesyonel çalışmalar için Adobe ve Microsoft ürünleri tercih edilebilir. Uzmanlar, farklı editörler kullanılarak önce deneyim sahibi olunmasını, sonrasında programcının ya da tasarımcının kullanma stili ve iş akışına göre editör seçimi yapmasını önermektedir.

HTML dilini kullanarak web sayfaları yapabilmek için temel etiketleri, başlıklar, paragraflar ve satır atlama etiketlerini bilmek gerekmektedir. Şimdi en temel etiketleri kullanarak ilk web sayfamızı hazırlayalım.

Tablo 1.2. HTML ile web sayfası

HTML Etiketleri	Web Sayfası Görüntüsü
<pre><html> <head> <title>ilk sayfam</title> </head> <body> Merhaba, ilk web sayfam! </body> </html></pre>	

Yukarıda görmüş olduğunuz kod bloğu bir web sayfasının temel HTML yapısıdır. HTML dokümanlarındaki ilk etiket `<html>` etiketidir. Bu `<html>` etiketi tarayıcılara bir HTML dosyasının çalıştırılacağını söyler. HTML dokümanlarındaki son etiket yine `</html>`'dir. Bu da tarayıcıya HTML sayfalarının sonunun geldiğini belirtir. Bütün diğer kodlar bu iki etiket arasına yazılır. Diğer etiketlerle ilgili bilgiler ise şöyle sıralanabilir:

- `<head>` ve `</head>` etiketleri sayfaların başlık bölümüyle ilgili bilgileri içerir ve tarayıcılarda görüntülenmemektedir.
- `<title>` ve `</title>` etiketleri sayfaların başlıklarını içerir ve bu başlıklar pencerelerin en üst tarafında görünülmektedir.
- `<body>` ve `</body>` tarayıcılarda görüntülenen bölümler bu etiketler arasında kalan bölümde verilmektedir. Bundan sonraki bölümde genelde bu etiket içine yazılan HTML etiketleri anlatılacaktır.
- HTML etiketleri genellikle parametreleriyle beraber kullanılır. Parametreler, HTML etiketlerine, farklı özellikler eklenmesini sağlar. Parametreler isim ve değer yazılarak kullanılır, örneğin `name="kimlik"`.



HTML sayfaları `<html>` ile başlayıp `</html>` ile biten etiketlerden oluşur.

Bireysel Etkinlik

- Bilgisayarınızdaki herhangi bir kelime editörü (not defteri ya da Notepad++ olabilir) ile Tablo 1.2.'deki kodları yazıp aynı klasöre uzantısı .html veya .htm olarak kaydedin. Ardından bilgisayarınızdaki tarayıcılar ile HTML sayfanızı görüntüleyin.

HTML'İN TEMEL ETİKETLERİ

HTML ile web sayfaları tasarlarken onlarca etiket kullanılmaktadır. Her sürümde bu etiketlere yenileri dâhil edilmiştir. Şimdi en yaygın bilinen etiketler ve web sayfasındaki görünümleri anlatılacaktır.

Meta etiketi web sayfalarının bilgi etiketi olarak kullanılır. Bu etiket, sayfayı ziyaret edenlere sayfanın yapısı ve içeriği hakkında bilgiler vermek için kullanılmaktadır. Bu etiket HTML kodlarında *head* etiketleri arasında yer alır ve tarayıcıda gözükmeyez. Meta etiketleri sayfaların açıklama (description), anahtar kelimeler (keywords), sayfa yazarı (author), son değiştirilme tarihi (last modified) ve diğer meta verilerini vermek için kullanılır. Ayrıca, bu etiketten gelen bilgilerle internet tarayıcıları sayfa yüklenirken nasıl görüntüleneceğine karar verir, arama motorları da indexleme ve anahtar kelimelerini alırlar. Yeni nesil arama motorları meta etiketlerden gelen bilgilerin yanında sayfaların gövde kısmındaki bilgileri de dikkate alarak arama yaparlar. Meta etiketlerinin kullanımına yönelik bazı örnekler aşağıda verilmiştir:

- Web sayfasının açıklama verisi için kullanılması gereken meta etiket kullanımı: `<meta name="description" content="HTML Dersleri"/>`
- Web sayfasının anahtar kelime verisi için kullanılması gereken meta etiket kullanımı: `<meta name="keywords" content="HTML ,CSS, Javascript"/>`
- Web sayfasının yazar verisi için kullanılması gereken meta etiket kullanımı: `<meta name="author" content="Atatürk Üniversitesi"/>`
- Web sayfasının içeriği görüntülenirken Türkçe karakter sorunu yaşamamak için kullanılması gereken meta etiketinin kullanımı: `<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-9"/>`

Başlık etiketi altı adettir, bunlar sırasıyla; `<h1>` , `<h2>` , `<h3>` , `<h4>` , `<h5>` ve `<h6>`'dır. Bunlardan en küçük olanı `<h6>`, en büyük olanı `<h1>`' dir.



HTML çok sayıda etiketten oluşmaktadır.

Her sürümde yeni etiketler eklenmektedir.

Tablo 1.3. Başlık Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<code><h1>İlk başlığım</h1></code>	
<code><h2>İlk başlığım</h2></code>	
<code><h3>İlk başlığım</h3></code>	
<code><h4>İlk başlığım</h4></code>	
<code><h5>İlk başlığım</h5></code>	
<code><h6>İlk başlığım</h6></code>	

HTML'de *paragraflar* `<p>` etiketi ile belirtilir.

Tablo 1.4. Paragraf Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<pre><p>Bu bir paragraf</p> <p>Bu da başka bir paragraf</p></pre>	 <p>Bu bir paragraf</p> <p>Bu da başka bir paragraf</p>

Satır atlamak için `
` etiketi kullanılır. Diğer yandan bu etiketin yeni bir paragraf oluşturmadığını da unutmamak gereklidir. Ayrıca diğer etiketlerden farklı olarak sadece başlangıç etiketi ile kullanmak yeterlidir.

Tablo 1.5. Satır Atlama Etiketi

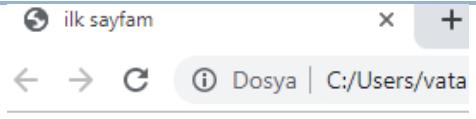
HTML Etiketleri	Web Sayfası Görüntüsü
<pre><p>Bu
birden fazla
satırдан oluşan paragraftır.</p></pre>	 <p>Bu birden fazla satırдан oluşan paragraftır.</p>

Yorum satırları, HTML kaynak kodu içerisinde programcılar yazdıkları satırların veya blokların ne amaçla yazıldığını anlatmak için kullanılır. Bu etiketin içeriği, tarayıcılarda görüntülenirken dikkate alınmazlar

`<!-- Bu bir açıklama satırıdır-->`

Sayfalarda alıntılar yapılırken, herhangi bir eserden, kitaptan veya kişiden alınacak sözler, metinler veya resimler *blockquote* (`<blockquote> </blockquote>`) etiketi içerisinde alınarak verilir.

Tablo 1.6. Blockquote Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<pre>Mustafa Kemal Atatürk diyor ki: <blockquote> TÜRK, ÖĞÜN, ÇALIŞ, GÜVEN..!</blockquote></pre>	 <p>Mustafa Kemal Atatürk diyor ki:</p> <p>TÜRK, ÖĞÜN, ÇALIŞ, GÜVEN..!</p>



İyi bir programcı yazdığı kod satırlarının aralarına açıklama satırları ekleyerek bilgilendirme yapar.



Bireysel Etkinlik

- Adınız soyadınızı başlık (örn: h1), hobilerinizi de paragraf etiketi (örn: <p>) kullanarak HTML sayfası hazırlayıp kayıt edin ve herhangi bir tarayıcıyla hazırladığınız sayfayı görüntüleyin.

HTML'de metinlerle ilgili çok sayıda etiket vardır. Aşağıdaki tabloda metinlerle ilgili kullanılan en yaygın etiketlerden bazıları örnek uygulamalarıyla verilmiştir.

Tablo 1.7. Metinlerle İlgili En Yaygın Etiketler

Etiket	Açıklama	Örnek Uygulama
.. 	Kalın ifade	Metin
<i>.. </i>	İtalik ifade	<i>Metin</i>
<u>.. </u>	Altı çizgili ifade	<u>Metin</u>
^{..}	Üst simge	X ²
_{..}	Alt simge	H ₂
 	Güçlü vurgu, kalın (bold) metin olarak gözükmür.	Metin
<hr>	Yatay çizgi	_____



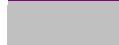
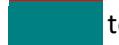
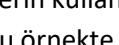
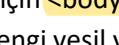
Örnek

- ve etiketleri birbirinin yerine kullanılabilir.

HTML'de renk kodları bir diyez (#) işaretini ile başlar ve daha sonra altı basamaktan oluşan bölüm gelir. Örn:
#FFFFFF

HTML sayfalarında *renkler* İngilizce adlarıyla kullanılabildiği gibi 16'lık sayı düzenindeki "hexadecimal" değerleriyle de kullanılabilir. HTML'deki *renkler* hexadecimal değerler ile RGB (kırmızı yeşil mavi) kombinasyonu ile gösterirler. Bu kombinasyonu seçerken en düşük değer 0 (hex #00) ve en yüksek değer de 255'tir (hex #FF). Web sayfalarında sıkılıkla kullanılan ve birçok tarayıcının desteklediği 16 renk ve isimleri (İngilizce olarak da) aşağıdakilerdir:

Tablo 1.8. Renkler Hakkında Bilgiler

Renk	Renk adı	Renk	Renk adı
	aqua (turkuvaz)		black (siyah)
	blue (mavi)		fuchsia (pembe)
	gray (gri)		green (yeşil)
	lime (fıstık yeşili)		maron (kestane)
	navy (lacivert)		olive (zeytin yeşili)
	purple (mor)		red (kırmızı)
	silver (gümüş)		teal (deniz mavisi)
	white (beyaz)		yellow (sarı)

Renklerin kullanımına örnek olması için `<body bgcolor="green">` etiket satırı verilebilir. Bu örnekte sayfanın arka plan rengi yeşil yapılır. Hexadecimal karşılığı da kullanılabilir, `<body bgcolor="#008000">` aynı sonucu verir.

`<body>` etiketi bilindiği üzere arka planları belirlemek için parametreler almaktadır. Bu parametrelere örnek olarak "bgcolor" ve "background" verilebilir. Sayfalara bgcolor parametresi ile arka plan rengi, background parametresi ile de arka plan resmi verilebilir.

```
<body bgcolor="black">
<body background="bg.jpg">
```

`Font` etiketi de renklerle çok sık kullanılan başka bir etikettir. Bu etiketin parametrelerinin alacağı değerler ile metinlerin renk, yazı tipi, boyut gibi özelliklerinin değiştirilmesi sağlanır.

```
<font face="...." size="...." color="...." > </font>
```

Font etiketinin aldığı bu parametrelerin özellikleri şöyle özetlenebilir:

- `face`= yazı tipinin adı (Times New Roman, Arial, Tahoma, Verdana gibi)
- `size`= metnin büyülügü (1-7 arası)
- `color`= metnin rengi (blue, black gibi renklerin İngilizce karşılığı ya da RGB renk değeri)



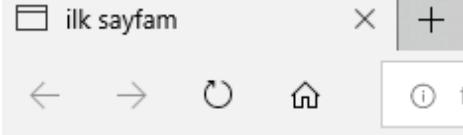
HTML'de font etiketi 3 parametre alır: `face`, `size`, `color`.

Tablo 1.9. Font Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<pre> MEVSİMLER
 İlkbahar
 Yaz
 Sonbahar
 Kış</pre>	 <p>MEVSİMLER İlkbahar Yaz Sonbahar Kış</p>

Linkler ... etiketi içinde, href="" komutuyla kullanılır. Target parametresi ile de sayfanın nerede açılacağı belirtilebilir. Target parametresi kullanılırken; "_blank" değeri yeni bir sayfa olarak açılacaksa, "_self" değeri aynı pencere içinde açılacaksa, "_top" değeri de aynı pencere içerisinde üstten itibaren açılacaksa verilir. Eğer özel kodlama kullanılmıyor ise (örneğin CSS kullanılmıyor ise) linklerin altı çizgili ve mavi renkte görünürlər.

Tablo 1.10. Bağlantı (link) Etiketleri

HTML Etiketleri	Web Sayfası Görüntüsü
<pre> Atatürk Üniversitesi</pre>	 <p>ilk sayfam</p> <p>← → ⌂ ⌂ Atatürk Üniversitesi</p>

Aynı sayfa içinde bağlantıyı diğer ifadeye link kurmak istersek de yine etiketi kullanılabilir. Başka bir ifadeyle bu etiket ile aynı sayfa içinde belirlenmiş başka bir satır ya da paragrafa direkt geçiş yapılabilir.



etiketileyi aynı sayfa içinde de başka bir satırda ya da paragrafa bağlantı verilebilir.

Bağlantı başlığı aşağıdaki gibi belirlenir.

Bağlantılar

Sayfanın herhangi bir yerinde yukarıda belirtilen bağlantıya aşağıdaki gibi ulaşabiliriz.

[Bağlantılara git](#link)


Bireysel Etkinlik

- Güncel gazetelerin isimlerini tek bir sayfada olacak şekilde listeyelin ve her bir gazete ismi seçilince ilgili gazetenin web sayfasına gidecek şekilde bağlantıya dönüştürün.

HTML'de bir diğer etiket ise **çerçeve (frame)** dir. Çerçeve etiketiyle aynı tarayıcı penceresinde birden fazla HTML sayfası görüntülenebilir. Tarayıcılarında her HTML sayfası bir **çerçeve** olarak görülür ve her çerçeve birbirinden bağımsız olarak çalışır. Bu etiket ile ilgili bilgiler özetlenecek olursa;

- **<frameset>** ile pencerenin hangi formatta çerçevelere ayrılacağı belirlenir.
- **<frameset>** kullanılırken satır veya sütunlar belirlenir.
- **<frame>** etiketi ile de çerçevelerde hangi HTML sayfasının görüntüleneceği belirtilir.

Aşağıda verilen örnekte 2 sütundan oluşan çerçeveli bir sayfa bulunmaktadır. İlk çerçeve için ayrılan pencere genişliği %30'dur. İkincisine ise %70'i ayrılmıştır. "sol.html" birinci sütuna, "sag.html" ise ikinci sütuna yerleştirilmiştir:

Tablo 1.11. Çerçeve (Frameset) Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<pre><frameset cols="30%,70%"> <frame src="sol.html"> <frame src="sag.html"> </frameset></pre>	

Aşağıdaki tabloda ise sol ve sağ olarak ikiye bölünen pencerelerin web sayfaları ve içeriği verilmektedir.

Tablo 1.12. Çerçevelerin HTML Etiketleri

sol.html	sag.html
<pre><html> <head> <title>çerçeve</title> </head> <body> sol çerçeve </body> </html></pre>	<pre><html> <head> <title>çerçeve</title> </head> <body> sağ çerçeve </body> </html></pre>

Web sayfalarında istenilen yere istenilen genişlikte bir çerçeve yerleştirmek için **iframe** kullanılmaktadır. Aşağıda örnek bir iframe kod satırı verilmektedir.

```
<html>
  <iframe src="test.html" width="250" height="400" scrolling="yes">
</html>
```

HTML'de **tablolar** <table>....</table> etiketleri arasında oluşturulur. <table> etiketinden sonra <tr> ve <td> etiketleri kullanılır. Her satır oluşturulması için <tr>, her hücre oluşturulması için <td> etiketi yazılır.



Örnek

- frame etiketi, yerini iframe etiketine bırakmaya başlamıştır.

Tablo 1.13. Tablo Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<pre><table border="1"> <tr><td>1. hücre</td> <td>2. hücre</td> <td>3. hücre</td> </tr> <tr><td>4. hücre</td> <td>5. hücre</td> <td>6. hücre</td> </tr> </table></pre>	

Table etiketinin aldığı çok sayıda parametre vardır. Örnek bir tablo satırı şu parametreleri içerebilir:

<table border="" cellspacing="" cellpadding="" height="" width="" bgcolor="" align="" valign="">

- **border:** Çerçevenin kalınlık değeri için kullanılır. border="0" şeklinde olursa tabloda çerçeve bulunmaz, 1 veya üzeri rakamlar kullanılırsa veya rakamlar arttırdıkça çerçeveyenin kalınlığı da artmış olur.
- **cellspacing, cellpadding:** Hücrelerin içeriğinin satır ve sütunların uzaklığını cellspacing, sınırlara olan uzaklığa içinse cellpadding parametresi kullanılır.
- **bgcolor:** <table bgcolor="blue"> olarak kullanılırsa bütün tablo ya da <td bgcolor="blue"> olarak sadece tek bir hücre renklendirilir.
- **align:** Hücre içeriğinin yatay konumunu belirler ve "right, left, center" seçenekleri ile kullanılır.
- **valign:** Hücre içeriğinin düşey konumunu belirler ve seçenekleri "top, bottom, middle" (üst, alt, orta)'dır.
- **rowspan, colspan:** Aynı sütundaki hücreleri birleştirmek için rowspan, benzer olarak aynı satırdaki hücreleri birleştirmek için colspan tercih edilir.

Tabloları detaylandırmak istersek aşağıdaki farklı özellikleri de tablolara ekleyebiliriz.

- <thead>.....</thead> başlık kısmını belirtir.

- <th>.....</th> başlıklı her bir alanı tanımlar.
- <tbody>.....</tbody> içerik bölümünü ayırrır.
- <tfoot>.....</tfoot> alt bölümü ayırrır.

Tablo 1.14. Tablo Etiketi Örneği

Tablo oluştururken bir sütuna hiçbir şey yazmadan boş bırakırsak sütun görünmeyecektir.

HTML Etiketleri			Web Sayfası Görüntüsü		
<pre><table border=1> <thead> <tr> <th>No</th> <th>Ad</th> <th>Soyad</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Ali</td> <td>Tatlı</td> </tr> <tr> <td>2</td> <td>Hasan</td> <td>Koç</td> </tr> <tr> <td>3</td> <td>Hüseyin</td> <td>Gül</td> </tr> </tbody> <tfoot> <tr> <td>Toplam</td> <td>---</td> <td>---</td> </tr> </tfoot> </table></pre>					

Yukarıdaki örnekte satır ve sütunlardan oluşan bir tablo örneği verilmiştir. Aşağıdaki örnekte ise hücreleri birleştirmek için *colspan*, aynı sütundaki hücreleri birleştirmek için de *rowspan* parametreleri kullanılmıştır.

Tablo 1.15. Table Etiketinin Rowspan ve Colspan Parametreleri

HTML Etiketleri	Web Sayfası Görüntüsü						
<pre><table border="1"> <tr><td>1. hücre</td><td>2. hücre</td><td>3. hücre</td></tr> <tr><td rowspan="2">4. hücre</td><td colspan="2">5. hücre</td></tr> <tr><td colspan="2">6. hücre</td></tr> </table></pre>	<table border="1"> <tr> <td>1. hücre</td> <td>2. hücre</td> <td>3. hücre</td> </tr> <tr> <td>4. hücre</td> <td>5. hücre</td> <td>6. hücre</td> </tr> </table>	1. hücre	2. hücre	3. hücre	4. hücre	5. hücre	6. hücre
1. hücre	2. hücre	3. hücre					
4. hücre	5. hücre	6. hücre					

HTML bize üç tip liste imkânı sunuyor: sıralı, sırasız, tanımlamalı.



HTML'de bir diğer etiket, **Liste** etiketidir ve 3 çeşidi vardır: **Sıralı, sırasız ve tanımlamalı**. Aşağıdaki tabloda en yaygın olan kullanım şekillerinden sıralı ve sırasız liste etiketine örnek verilmiştir. **Sıralı listeler** rakam, harf veya her ikisini iç içe kullanarak liste oluşturmayı sağlarken **sırasız listeler** rakam/harf yerine madde imleri koyarak liste oluşturmayı sağlar. Tanımlama listeleri ise uzun metinlerde okumayı kolaylaştıran etikettir.

Tablo 1.16. Sıralı ve Sırasız Etiketler

HTML Etiketleri	Web Sayfası Görüntüsü		
<pre>Ülkeler: Türkiye Almanya Fransa Şehirler: İstanbul Ankara Erzurum </pre>	<table border="1"> <tr> <td> Ülkeler: 1. Türkiye 2. Almanya 3. Fransa </td> <td> Şehirler: • İstanbul • Ankara • Erzurum </td> </tr> </table>	Ülkeler: 1. Türkiye 2. Almanya 3. Fransa	Şehirler: • İstanbul • Ankara • Erzurum
Ülkeler: 1. Türkiye 2. Almanya 3. Fransa	Şehirler: • İstanbul • Ankara • Erzurum		

Sıralı ve Sıralamasız listelerde, liste öğelerini belirtmek için farklı semboller kullanılabilir. Aşağıdaki tabloda atanabilecek değerler ve açıklamaları verilmiştir.

Tablo 1.17. Sıralı ve Sırasız Liste Çeşitleri

Sıralamasız liste (<ul style="list-style-type: none">) türü	Açıklama
disc	İçi dolu daire
circle	İçi boş daire
square	İçi dolu ya da boş kare
Sıralı liste (<ol style="list-style-type: none">) türü	Açıklama
1	Onluk tabanda numaralama (1,2,3,4,...)
i	Küçük rakamlarla i, ii, iii, iv gibi Romen sayıları
I	Büyük rakamlarla I, II, III, IV gibi Romen sayıları
a	Küçük harflerle a, b, c gibi
A	Büyük harflerle A, B, C gibi

Sıralı listede maddeleri
1 2 3; a b c; A B C; I II III;
i ii iii şeklinde
sıralayabiliriz.



Tanımlı listeler başlık oluşturmak ve o başlıkların altında içerik paylaşmak için kullanılır. **<dt> ... </dt>** etiketi ile başlıklar, **<dd> ... </dd>** etiketi ile de başlıklara ait içerikler belirtilmektedir.

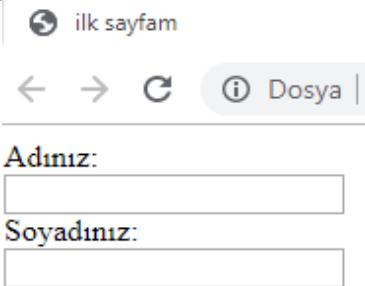
Tablo 1.18. Tanımlı Liste Etiketi

HTML Etiketleri
<pre><dl> <dt>Atatürk Üniversitesi 1950lı Yıllar <dd>17 Kasım 1958'de Ziraat ve Fen- Edebiyat Fakülteleri ile öğretime başladığında 12'si kız olmak üzere toplam 135 öğrencisi vardı. </dd></dt> <dt>Atatürk Üniversitesi 1960'lı Yıllar <dd>1966 Şubat'ında Tıp Fakültesi'nin açılması ile üniversitenin fakülte sayısı üçe yükseldi. Ardından da Fen-Edebiyat Fakültesi bünyesindeki Fen, Edebiyat ve İktisat-İşletme bölümlerinin ayrı birer fakülte haline getirilmesi çalışmaları 1968 Kasım'ında sonuçlandı. </dd></dt> <dt>Atatürk Üniversitesi 1970li Yıllar <dd>Yıllar sonra Atatürk Üniversitesi'ne "üniversite kuran üniversite" unvanını getirecek ilk gelişme 1971'de yaşanacaktır.</dd></dt> </dl></pre>
Web Sayfası Görüntüsü
<p>Atatürk Üniversitesi 1950lı Yıllar</p> <p>17 Kasım 1958'de Ziraat ve Fen- Edebiyat Fakülteleri ile öğretime başladığında 12'si kız olmak üzere toplam 135 öğrencisi vardı.</p> <p>Atatürk Üniversitesi 1960lı Yıllar</p> <p>1966 Şubat'ında Tıp Fakültesi'nin açılması ile üniversitenin fakülte sayısı üçe yükseldi. Ardından da Fen-Edebiyat Fakültesi bünyesindeki Fen, Edebiyat ve İktisat-İşletme bölümlerinin ayrı birer fakülte haline getirilmesi çalışmaları 1968 Kasım'ında sonuçlandı.</p> <p>Atatürk Üniversitesi 1970li Yıllar</p> <p>Yıllar sonra Atatürk Üniversitesi'ne "üniversite kuran üniversite" unvanını getirecek ilk gelişme 1971'de yaşanacaktı.</p>

Formlar **<form>...</form>** etiketleri arasında yazılır. Form değerleri **action** parametresinin içine yazılan dosyaye veri olarak gönderilir. Formlar sayesinde çeşitli anketler ve geri bildirim içeren sayfalar hazırlanabilir. Giriş (Input), Seç

butonları (Radio Buttons), Kontrol kutuları (Checkboxes), Çok satırlı metin giriş alanı (textarea) kullanılan en yaygın form etiketleridir. Öncelikle aşağıdaki gibi basit bir form örneği yapalım:

Tablo 1.19. Form Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<pre><form> Adınız:
 <input type="text" name="adiniz">
 Soyadınız:
 <input type="text" name="soyadiniz"> </form></pre>	



Web sayfalarında etkileşim olması için form etiketleri kullanmak gereklidir.

Yukarıdaki örnekte, basit form elemanlarıyla iki adet (Adınız, Soyadınız) metin girişi yapılan bir form hazırlanmıştır. Bu form elemanlarını daha detaylandırmak mümkündür. Bunun için aşağıdaki örnek verilebilir.

Tablo 1.20. Form Elemanları

HTML Etiketleri
<pre><form name="ornek" action="veri.php" method="post"> Ad/Soyad : <input type="text" size="20" name="ad">
 Doğum yeri : <input type="text" size="20" name="dogumyeriniz" >
 Doğum tarihi : <input type="text" size="10" name="dogumtarihiniz" >
 Cinsiyet : <input type="radio" name="cins" value="erkek" > Erkek <input type="radio" name="cins" value="kadın" > Bayan
 Hobiler:
 <input type="checkbox" name="muzik" >Müzik dinlemek
 <input type="checkbox" name="kitap" >Kitap okumak
 <input type="checkbox" name="diger">Diğer :
 <textarea rows="4" cols="30" name="diger"></textarea>
 <input type="submit" value="GÖNDER"> <input type="reset" value="SİL"> </form></pre>

Web Sayfası Görüntüsü

Ay/Soyad :

Doğum yeri :

Doğum tarihi :

Cinsiyet : Erkek Bayan

Hobiler:

Müzik dinlemek

Kitap okumak

Diğer :

GÖNDER SIL

Yukarıdaki form kod satırlarını daha detaylı inceleyelim:

- **action:** Formun gönderileceği dosyayı belirtir.
- **method:** Formdaki bilgilerin hangi yöntemle (post/get) gönderileceğini seçmek için kullanılır.
- **type="text":** Bir satırlık bir giriş alanı açar.
- **type="checkbox":** Çok seçenekli soru hazırlanır.
- **type="radio":** Tek seçenekli bir sorunun tek cevabını alır.
- **type="submit":** Formu action'la belirtilen dosyaya gönderen butondur.
- **<textarea rows="" cols="":** Çok satırlı metin alanları oluşturmak için kullanılır.



Method özelliği, form verilerinin gönderilme yöntemini belirtir. Veriler GET ve POST olmak üzere iki şekilde gönderilebilir.

Select ve option ise seçimlik listeler oluşturulurken kullanılan etiketlerdir.

Option etiketi ile belirtilen listenin her bir elamanı oluşturulur ve fareyle seçilen bu elemanlardan biri select etiketinin değeri hâline gelir. Aşağıdaki örnekte Alfa Romeo select etiketinin değeri olmuştur.

Tablo 1.21. Select ve Option Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<pre><select name="Arabalar"> <option>Alfa Romeo</option> <option>BMW</option> <option>Mercedes</option> </select></pre>	<p>Alfa Romeo BMW Mercedes</p>



Resim üzerinden başka bir belgeye bağlantı vermek için `` etiketi `<a>` etiketi içerisinde kullanılmalıdır.

HTML sayfalarında **resimler** `` etiketi ile belirtilir. ** etiketi** özelliği gereği parametreler ile kullanılır ve bir bitiş etiketi bulundurmaz.

Sayfalarda resim görüntüleyebilmek için "src" parametresinin kullanılması zorunludur. Src, İngilizce source kelimesinin kısaltması, yani kaynak anlamına gelir. Buraya görüntülemek istenilen resim dosyasının yolu yazılır. "Alt" parametresi alternatif metin anlamına gelir. Başka bir ifadeyle, ilgili resmin üzerine fare gelip bir süre beklentiği zaman görünen metindir.

Tablo 1.22. img Etiketi

HTML Etiketleri	Web Sayfası görüntüsü
<code></code>	<p>The screenshot shows a web browser window with a toolbar at the top. Below the toolbar, there is a search bar with the placeholder "file:///". In the center of the browser, there is a circular logo of Ankara University, which is blue with a golden peacock design in the center.</p>

Aşağıda `` etiketinin diğer parametreleri verilmiştir.

Tablo 1.23. img Etiketinin Parametreleri

Parametre	Açıklama
<code>width</code>	Pikseltan resmin genişliğini belirtir.
<code>height</code>	Resmin yüksekliği belirtilir.
<code>border</code>	Resmin çevresindeki çizginin kalınlığı belirtilir.
<code>align</code>	Resmin yatay konumunu belirtir; left, right, center değerlerini alır.

Bu hafta HTML diline giriş yapılmıştır. Tarihçesinden başlanarak en yaygın kullanılan etiketler anlatılmıştır. Kitabın bundan sonraki bölümlerinde HTML dilinin yeni sürümlerinden bahsedilecektir. HTML dilini öğrenebilmeniz için her bölümde verilen örnekleri kendiniz yazarak denemeler yapabilirsiniz.



Bireysel Etkinlik

- Kendi öz geçmişinizi içeren bir web sayfası oluşturun. Bu sayfayı tablo etiketi ile tasarlayabilirsiniz. Sayfanın arka planını farklı renk yapın, resminizi ekleyin, öğrenim gördüğünüz okulları, ilgi alanlarınızı tablonun ilgili satır ve sütunlarına yerleştirin.



Ozet

•HTML NEDİR?

- HTML'in, İsviçre'deki CERN Araştırma Enstitüsünde icat edildiği söylenmektedir.
- HTML'in ilk sürümü 1990larının başlarında yayınlanmıştır ve az sayıda (18 adet) HTML etiketi içermektedir. İlk sürümünden günümüze, HTML'nin her yeni sürümü yeni etiketleri ve parametreleri beraberinde getirir.
- HTML ile görsel, yazı ve video gibi içeriklerin web sayfasına konumlandırılması sağlanmaktadır. Oluşturulan içeriklerin web sitelerinde görüntülenmesi sağlanmaktadır.
- HTML dili resmî bir web standartı olarak kabul edilmektedir. HTML özellikleri World Wide Web Consortium (W3C) tarafından denetlenmekte ve geliştirilmektedir.

•HTML'İN TEMEL ÖZELLİKLERİ

- HTML kod satırları genel olarak <etiket></etiket> şeklinde tanımlanmış etiketler arasında yazılır.
- Bu etiketler 2 karakter ile sınırlanır: < ve >
- Etiketler çift olarak kullanılır. Örn: <u> Bu metnin altı çizgilidir. </u>
- İlk etiket başlama etiketi, ikincisi ise bitiş etiketidir.
- Başlama ve bitiş etiketleri arasında kalan her şey içerik olarak görülür.
- HTML etiketleri büyük/küçük harfe duyarlı değildir, örneğin
 ve
 aynı görevi görür.
- HTML dosyaları .html veya .htm uzantısıyla biten belgelerdir.
- HTML belgeleri, herhangi bir internet tarayıcısı (Google Chrome, Safari veya Mozilla Firefox) kullanılarak görüntülenebilir.
- HTML sayfası hazırlamak için not defteri, wordpad veya word dosyası yeterli olacaktır. Ayrıca, profesyonel programlardan Adobe Dreamweaver, Microsoft Expression Web gibi araçlar kullanarak da daha kolay ve hızlı HTML sayfaları oluşturulabilir.
- HTML, CSS ve JavaScript aynı anda kullanıldığından daha görsel ve dinamik web safaları hazırlanabilir.

•HTML'İN TEMEL ETİKETLERİ

- <html></html> etiketi her bir HTML sayfasını başlatıp kapatın etikettir.
- <head></head> etiketi, sayfa başlığı ve karakter sınırı gibi meta bilgilerini içerir.
- <title></title> sayfaların başlık bilgisi bu etiket altında yer alır.
- <meta></meta> etiketi web istemcisine ve arama motorlarına web sayfası hakkında bilgi verir.
- <body></body> etiketi sayfada görünen tüm içeriği kapsar.
- HTML'de başlıklar 6 seviyeye sahiptirler. <h1></h1>'den <h2></h2>'ye kadardır, h1 en yüksek seviye başlık iken, h6 ise en küçüğdür.
- HTML'de çok sayıda metin düzenleyici etiket vardır. Bu etiket ile metinlerin şekli, tipi, kalınlığı gibi özellikler değiştirilebilir. Paragraflar <p></p> etiketile kapatılır. Satır atlama
 etiketidir. Yatay çizgi <hr> etiketidir. Bunların yanında; kalın, <i>eğik</i>, <s>üzeri çizili</s>, <u>altı çizili</u>, <small>küçük</small>, <big>büyük</big>, <tt>daklılo yazısı</tt> etiketleri vardır.



Özet(devamı)

- HTML etiketleri parametreler arak ek özelliklere sahip olurlar. Örneğin paragraf etiketi align parametresiyle ilgili paragrafin görünüşünü değiştirebilir. align = "left": Paragrafi sola dayalı olarak yazar, align = "right" : Paragrafi sağa dayalı olarak yazar. align = "center": Paragrafi ortalar. align = "justify": Paragrafi sola ve sağa dayalı olarak yazabilmek için sözcüklerin ara boşluklarını değiştirir.
- HTML'de link etiketi () ile başka sayfalara bağlantılar kurulabilir, image etiketi () ile sayfalara resimler eklenebilir. Bağlantılar genellikle öntanımlı olarak altı çizili ve mavi olarak görüntülenir.
- HTML'de elemanları belli bir sıraya göre liste mantığında oluşturmaya yarayan etiketler
,
 ve

etiketleridir.
- HTML sayfalarını daha organize etmek için tablo etiketi kullanılır. Üç temel etiketi vardır:

: hücreleri oluşturmak için.

- Çerçevevler , bir tarayıcı penceresini birden fazla pencereye bölüp her bir pencere içerisinde farklı içerikler gösterilmesini sağlar.
- HTML'de etkileşimli sayfalar yapmak mümkündür. Özellikle kullanıcıdan bilgi almak ve o bilgileri işlemek için formlar kullanılır (). Günümüzde, e-ticaret sitelerinden bankalara, anketlere kadar birçok içerik formları ile etkileşimli hâle getirilmektedir.
- HTML'de çok sayıda etiket vardır. Bu hafta en çok kullanılanları inceledik. Bu etiketlere birkaç tane daha örnek eklemek gerekirse alıntılar verilirken
> </blockquote>

 etiketleri kullanır. </acronym> kısaltmaların açılımı hakkında bilgi vermek için kullanılır.

DEĞERLENDİRME SORULARI

1. HTML'in tanımıyla ilgili aşağıdakilerden hangisi doğrudur?
 - a) Web sayfalarını oluşturmak için kullanılan standart işaretleme dilidir.
 - b) Sayfalar arası geçiş sağlayan format aracıdır.
 - c) Nesneleri bağlayan linktir.
 - d) ASP ve PHP benzeri programlama dilidir.
 - e) JavaScript ve CSS ile uyumlu değildir.

2. Html etiketleri aşağıdaki hangi karakterler arasında yazılır?
 - a) { }
 - b) < >
 - c) " "
 - d) # #
 - e) // //

3. Aşağıdakilerden hangisi alt satıra geçmek için kullanılır?
 - a) <p>
 - b)

 - c) <ur>
 - d) <bt>
 - e) <hr>

4. Aşağıdakilerden hangisi giriş alanı (textbox) için kullanılan HTML etiketidir?
 - a) <input type="text">
 - b) <input type="textfield">
 - c) <type="text">
 - d) <textbox>
 - e) <textbox type="text">

5. Aşağıdaki hangi HTML parametresi bir tablonun kenarlık kalınlığını belirlemek için kullanılır?
 - a) border
 - b) height
 - c) width
 - d) align
 - e) colspan

6. Aşağıda verilen bağlantı (link) satırlarından hangisi doğrudur?
 - a)
 - b) tıklayın
 - c) <href="www.atauni.edu.tr">
 - d) <a=" www.atauni.edu.tr">tıklayın
 - e) tıklayın
7. Web sayfasının arka zeminini kırmızı yapmak için aşağıdaki satırlardan hangisi kullanılır?
 - a) <body color="blue">.
 - b) <body set ="red">
 - c) <body bgcolor="red">
 - d) <body color="red">
 - e) <body backgroud="red">
8. Tabloya satır eklemek için hangi etiket kullanılır?
 - a) <td>satır ekle</td>
 - b) <th>satır ekle</th>
 - c) <tp>satır ekle</tp>
 - d) <tr>satır ekle</tr>
 - e) <table>satır ekle</table>
9. Sıralı liste oluşturmak için hangi etiket kullanılır?
 - a) <dl>
 - b) <dt>
 - c) <list>
 - d)
 - e)
10. Font etiketiyle beraber hangi parametre kullanılmaz?
 - a) face
 - b) size
 - c) width
 - d) color
 - e)

Cevap Anahtarı

1.a, 2.b, 3.b, 4.a, 5.a, 6.b, 7.c, 8.d, 9.e 10 -

YARARLANILAN KAYNAKLAR

HTML Tutorial. 7 Ağustos 2019 tarihinde

<https://www.w3schools.com/html/default.asp> adresinden erişildi.

HTML. 7 Ağustos 2019 tarihinde

<https://gelecegiyazanlar.turkcell.com.tr/konu/web-programlama/egitim/101> adresinden erişildi.

HTML Dersleri. 7 Ağustos 2019 tarihinde <http://www.htmldersleri.org> adresinden erişildi.

HTML5 İLE GELEN YENİLİKLER



- HTML5'in Temel Özellikleri
- HTML5 ile Gelen Yenilikler
- HTML5 ile Değişen Etiketler
- HTML5 ile Yeni Semantik-Anlamsal Etiketler
- HTML5 ile Yeni Form Elementleri

iÇİNDEKİLER



- Bu üniteyi çalıştıktan sonra;
 - HTML5'in doğuş hikâyesini açıklayabilecek,
 - HTML5 ile gelen yenilikleri sıralayabilecek,
 - HTML5 ile değişen etiketleri ayırt edebilecek,
 - HTML5 ile yeni anlamsal etiketlerini ve form elementlerini kullanabileceksiniz.

HEDEFLER

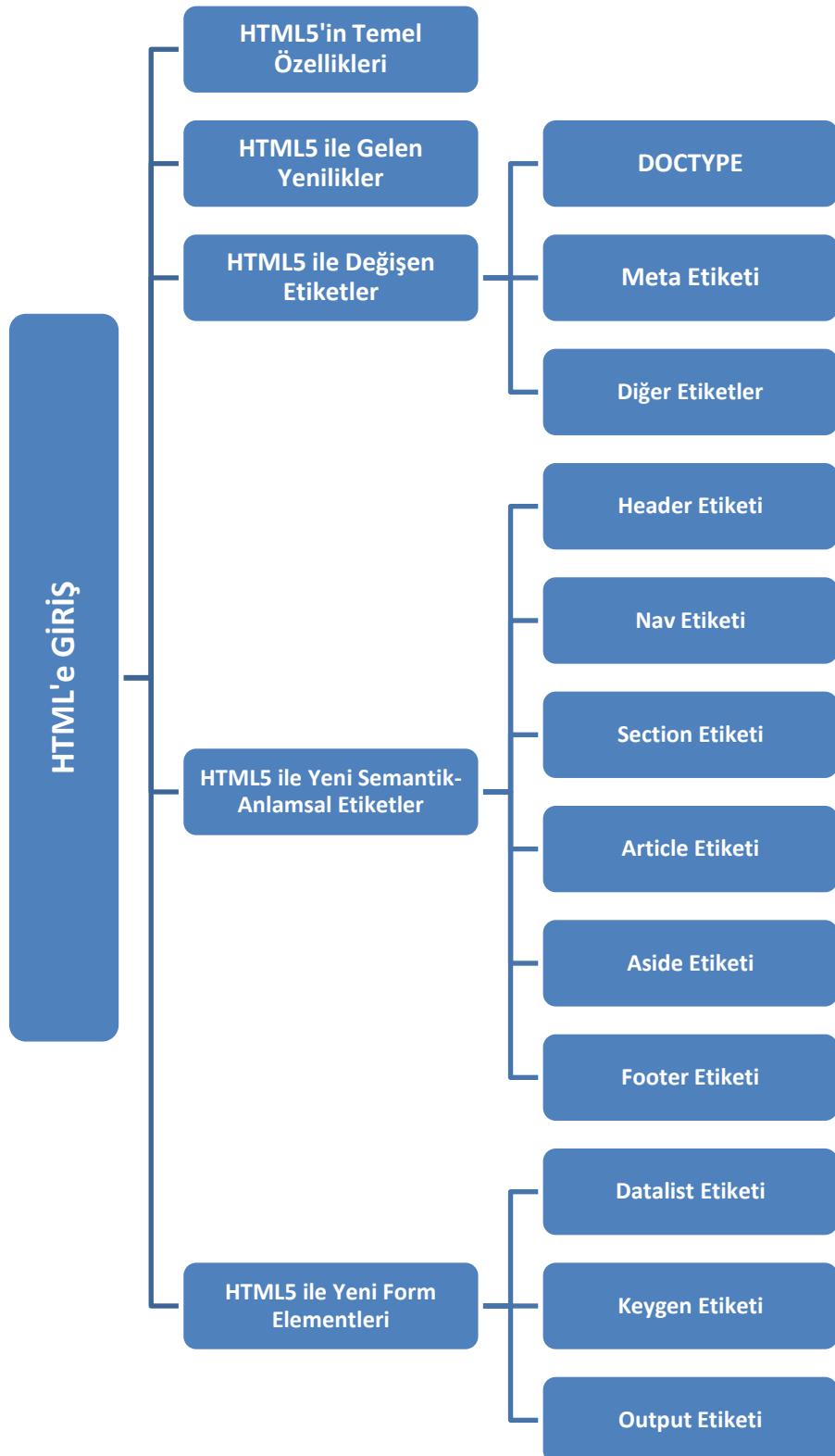


Atatürk Üniversitesi
Açıköğretim Fakültesi

INTERNET
PROGRAMCILIĞI II
Doç. Dr. İbrahim
ÇETİN

ÜNİTE

2



GİRİŞ

HTML5, HTML dilinin en son sürümüdür. Bu yeni sürüm ile HTML dili yeni stratejiler ve hedefler ortaya koyan ve farklı web tarayıcıları için standartizasyonu amaçlayan yeni nesil bir teknoloji olarak görülmektedir. Başka bir ifadeyle, HTML5 ile HTML dilinin işlevselligi, programlama ve sunum gücü çok daha fazla artmıştır. HTML5'e yeni eklenen yapısal etiketler sayesinde fazla kod yazmadan web sayfalarının görsel tasarımları yapılabılır, JavaScript ve CSS teknolojileriyle beraber daha etkileşimli bir şekilde projeler üretilebilmektedir. Artık zengin medya içerikleri ve interaktif web uygulamaları kolaylıkla ve hızlı bir şekilde geliştirilebilmekte ve web sayfalarına adapte edilebilmektedir.



HTML5, HTML dilinin son sürümüdür.

HTML5, daha önceki sürümlerin birçok özelliğini barındırmakla beraber yeni oluşturulan ve güncel ihtiyaçlara cevap verebilecek teknolojileri de içermektedir. *W3C (World Wide Web Consortium)* tarafından önceden duyurulmuş ve modern web tarayıcıları tarafından desteklenecek birçok teknoloji bu sürüm içerisine dâhil edilmiştir. Ayrıca W3C, HTML5 dilini geliştirmeye devam etmektedir. Bu açıdan ilerleyen zaman dilimlerinde bu sürümde de yeni eklentilerin olması beklenmektedir. Bu yeni sürümün diğer bir avantajı da HTML5'in duyurulması ile beraber önemi artan mobil teknolojilerinin de desteklenmeye başlanmasıdır.

HTML5'ten önce 1990'lı yıllarınlarında W3C tarafından HTML 4.01 duyurulmuştur. HTML'in bu sürümünden kısa bir süre sonra XML 1.0 yapısı yayınlandı. W3C, HTML dilini XML tabanlı yapmak için bu iki yapıyı birleştirdi ve 2001 yılında XHTML 1.0 olarak duyurmuştu. W3C, 2003 yılında duyurduğu XForms 1.0 (XHTML Extended Forms) bildirimi ile XHTML yapısını güçlendirmek istemiştir. W3C yazarlarının düşüncesi XHTML yapısının geliştirilmesi ve ek teknolojilerle desteklenmesi şeklindeydi (XHTML2 + XForms + SVG + MathML + RDFa). Bu yapı tarayıcı üreticileri tarafından kabul görmediği için 2004 yılında Apple, Firefox ve Opera tarayıcı üreticileri bir araya gelerek WHATWG (Web Hypertext Application Technology Working Group) isimli bir çalışma grubu oluşturdu. Aslında HTML5'in doğuş hikâyesi burada başlamış oldu.

2006 yılında W3C konsorsiyumu *XHTML* dilini geliştirmekten vazgeçip HTML5'in gelişimine katılacağını duyurmuştur. 2008 yılında W3C tarafından HTML5'in (First Public Working Draft) ilk çalışma taslağı duyuruldu. W3C konsorsiyumu 2009 yılında XHTML 2.0 çalışmalarını durduracağını açıklamıştır. Henüz HTML5 ile gelen özelliklerin tamamı bazı tarayıcılar tarafından desteklenmese de önumüzdeki günlerde web tasarımının vazgeçilmezi hâline geleceğine kesin gözüyle bakılmaktadır. Bunun nedeni ise daha önce web programlama dilleri ile yapılan işlerin HTML5 ile kolay şekilde yapılmıyor olması ve web tasarımındaki birçok yeni özelliğin HTML5 ile kullanılmaya başlanmasıdır. Bu bölümde HTML5'in temel özellikleri anlatılacak ve HTML5 ile gelen yeniliklerden bahsedilecektir.

HTML5'İN TEMEL ÖZELLİKLERİ



HTML5 ile HTML dilinin temel özelliklerinde değişimler olmuştur.

HTML, tek başına statik bir web sitesi oluşturmak için yeterlidir. Metin, resim, fotoğraf gibi öğeleri gösteren ve bunları birbirine bağlayan sayfalar arasında etkileşim yaratmak mümkündür. Fakat günümüzde programlama temelli dinamik web sayfaları beklenmekte, daha çok etkileşim talep edilmektedir. Artık grafik ve çoklu ortam desteği önemli hâle gelmiştir. Bu nedenle HTML dili üzerinde çalışılarak günümüz teknolojilerini destekleyen HTML5 sürümüne geçiş yapılmıştır. HTML5'in temel özelliklerini sıralayacak olursak;

- HTML4, XHTML ve diğer sürümlerle uyumlu olmakla beraber sadeleştirilmiştir.
- Daha fazla yapısal etiketlere, form elemanlarına ve yeni özelliklere sahiptir.
- Gelişmiş çoklu ortam desteği sunmaktadır. HTML5, eklentilere veya yazılımlara ihtiyaç duymadan (Flash gibi) çoklu ortam ve animasyon görüntülerini oynatabilir.
- Grafik işlemlerini destekleyen yeni elementleri barındırır.
- 2B ve 3B çizim desteği ile oyuncular yazılabilir.
- HTML5 kompleks web uygulamaları için, konum belirleme, sürükle bırak, yerel depolama gibi yeni API'ler sunar.
- *JavaScript ve CSS* teknolojilerinin daha etkili bir şekilde kullanımına imkân tanır. Başka bir ifadeyle HTML5'in yeni özellikleri HTML, CSS, DOM ve JavaScript üzerine kuruludur.
- HTML5 aynı zamanda cihazdan bağımsızdır.
- HTML5 çok platformlu mobil uygulamalar için de kullanılabilir.

Kısaca özetlersek, son yıllarda HTML5 teknolojilerinin kullanımının web sayfalarında yaygınlaştiği görülmektedir.



Örnek

- HTML5'i Google başta olmak üzere Facebook, Twitter, LinkedIn, Apple gibi büyük yazılım firmaları tercih etmektedir.

HTML5 İLE GELEN YENİLİKLER

Kitabın bundan sonraki bölümlerinde sırasıyla aşağıdaki başlıklar hâlinde HTML5 ile ilgili bilgiler verilecektir.

- HTML5 ile değişen etiketler
- HTML5 ile yeni semantik-anlamsal etiketler
- HTML5 ile yeni form etiketleri
- HTML5 ile yeni veri giriş tipleri
- HTML5 ile yeni form özellikleri
- HTML5 ile çoklu ortam elementleri

- HTML5 ile grafik elementleri
- HTML5 ile konum elementleri
- HTML5 ile sürükle bırak
- HTML5 ile gelen diğer API'ler

HTML5 ile Değişen Etiketler



HTML5 ile DOCTYPE, META gibi etiketler detay verilmeden kullanılmaktadır.

HTML5, HTML'in eski sürümlerinden bağımsız bir sürüm olmayıp daha önceden kullanılan standartlar hâlâ devamlılığını korumaktadır. Aslında HTML5 ile HTML'in önceki sürümlerine farklı ve yeni etiketler, elementler eklenerek geliştirilmiştir. Yeni teknolojilere geçmeden önce HTML'deki var olan bazı etiketlerin HTML5 ile değişikliklere uğradığından bahsetmek gereklidir. HTML5 ile ilgili ilk yenilik DOCTYPE alanına gelen değişiklik olmuştur. *DOCTYPE*, tarayıcılara dokümanın tipini belirtir. Örneğin, DOCTYPE'tan sonra yazacağımız ifade ile tarayıcı bu dokümanın tipini anlar. Web sayfaları için HTML ifadesi kullanılır. DOCTYPE, HTML'in daha önce sürümlerinde kullanılıyormasına karşın HTML5 ile değişikliğe gidilerek sadeleştirilmiştir. Aşağıdaki tabloda önceki sürümlerdeki ve HTML5 arasındaki farklar görülmektedir.

Tablo 2.1. DOCTYPE Alanındaki Değişim

Sürüm	DOCTYPE
XHTML 1.0	<!DOCTYPE html PUBLIC "-//W3C/DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
XHTML 1.0	<!DOCTYPE html PUBLIC "-//W3C/DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
HTML 4.01	<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
HTML 5	<!DOCTYPE html>

Gördüğü üzere HTML5'ten önce DOCTYPE verilirken; URL, versiyon numarası, sayfanın dili gibi bilgiler verilmektedir. HTML5'te ise önceki sürümlerdeki yapının kurulmasına gerek kalmamıştır. Bunun yerine daha basit kullanım getirilmiştir. Aşağıda en temel özelliklere sahip bir HTML5 sayfası görülmektedir.

Tablo 2.2. Örnek HTML5 Sayfası

```
<!DOCTYPE html>
<html> <head>
    <meta charset="UTF-8">
    <title>Başlık</title> </head>
    <body>
        Sayfa içeriği
    </body>
</html>
```



Bireysel Etkinlik

- Bilgisayarlarınızdaki herhangi bir kelime editörü (not defteri ya da Notepad++ olabilir) ile Tablo 2.2.'deki kodları yapıştırın klasöre uzantısı .html veya .htm olarak kaydedin. Ardından bilgisayarlarınızdaki tarayıcılar ile ilk HTML5 sayfanızı görüntüleyin.



Türkçe dil desteği için UTF-8 meta karakter seti kullanılmaktadır.

Değişikliğe gidilen bir diğer etiket ise **META** etiketidir. Bu etiket ile sayfalar hakkında bilgilerden karakter kodlamasına kadar birçok özellik belirtilmektedir. Örneğin, bu etiket ile bir web sayfasının hangi dile destek vereceğine karar verilmektedir. HTML 5'te yine meta etiketi kullanılmaktadır, fakat ayrıntıları azaltılmıştır. Aşağıda hem HTML4 hem de HTML5 ile web sayfamızı Türkçe olarak tasarlamış isek buna uygun bir karakter setinin nasıl kullanılacağına dair örnek verilmektedir.

Tablo 2.3. Meta Etiketindeki Değişim

Sürüm	META
HTML 4	<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
HTML 5	<meta charset="UTF-8">

HTML5'te web sayfası dilinin belirtilmesinin bir başka yöntemi ise aşağıdaki örnekte olduğu gibi html etiketinin **lang** parametresi "en" yani "English – İngilizce" olarak, "tr" ise "Türkçe" olarak belirtilebilmektedir. Birçok dil desteği benzer yöntemle desteklenmektedir.

Tablo 2.4. Lang Etiketi Kullanımı

<html lang="en">
<html lang="tr">

HTML5'te style ve script etiketlerinin kullanımında da değişikliğe gidilerek daha sade hâle getirilmiştir. Örnek yazım şekilleri aşağıdaki tabloda verilmiştir.

Tablo 2.5. Style ve Script kullanımı

HTML 4'de style ve script etiketleri	HTML5'te style ve script etiketleri
<style type="text/css"> css kodları </style>	<style> css kodları </style>
<script type="text/javascript"> kodlar </script>	<script> kodlar </script>

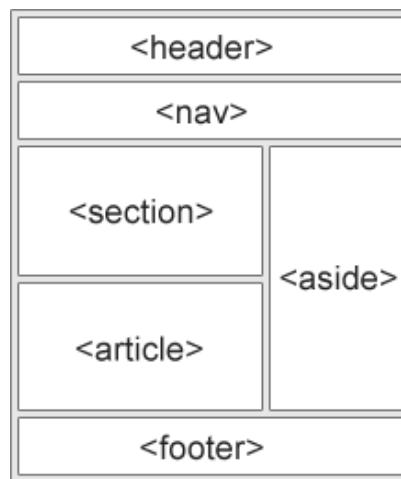
HTML5 ile Yeni Semantik-Anlamsal Etiketler

HTML 5 ile gelen yenilikler incelendiğinde karşımıza birçok yeni etiket çıkmaktadır. Bu etiketlerin özellikle web tasarımını yapanların işlerini kolaylaştırmak

üzere üretilmiş olduğu görülmektedir. Örneğin birçok web sayfası incelendiğinde şu şekilde kodlandığı görülmektedir:

- <div id="nav">,
- <div class="header">,
- <div id="footer">.

Başka bir ifadeyle **div** etiketleri kullanılarak web sayfaları böülümlere ayrılmaktadır. Web sayfaları üst, orta ve alt böümlerden oluşmaktadır. HTML5 bu sayfa tasarımlarını kolaylaştırmak üzere yeni çözümler sunmaktadır. Aşağıdaki yapıda görüleceği üzere çok sayıda yeni semantik-anlamsal etiket (<header>, <nav>, <section>, <article>, <aside>, <footer> gibi) gelmiştir.



Şekil 2.1. HTML5 Tasarım Şablonu



Örnek

- Şablonla örnek bir web sayfası olarak <https://www.atauni.edu.tr/>

HTML5 ile HTML4'deki bazı alışkanlıklar değişiklik göstermektedir ve daha basit ve sade kod satırlarıyla daha kolay tasarımlar yapılmaktadır. HTML5 ile gelen yeni etiketleri incelemeden önce HTML4 ile karşılaştırması aşağıdaki tabloda verilmektedir.

Tablo 2.6. HTML4 ve HTML5 Arasındaki Anlamsal Etiket Farlılıkları

HTML4	HTML5
<div id="header">	<header>
<div id="menu">	<nav>
<div id="content">	<section>
<div class="article">	<article>
<div id="footer">	<footer>

<header> etiketi, sayfanın başlığını temsil eder. Genellikle bu etiket sayfanın en üstünde yer alır ve bir logo, başlık ya da slogan benzeri metinleri içerebilir. Aşağıdaki örnekte ERZURUM başlığı *header* etiketi içinde verilmiştir.

Tablo 2.7. Header Etiketi

HTML5 Etiketleri
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>İlk HTML belgesi</title> </head> <body> <article> <header> <h1>ERZURUM</h1> </header> <p>Medeniyetler beşiği Anadolu'nun,
 medeniyet çeşitliliğiyle beslenen kadim bir toprak parçasıdır Erzurum.</p> </article> </body> </html></pre>
Web Sayfası Görüntüsü
ERZURUM
Medeniyetler beşiği Anadolu'nun, medeniyet çeşitliliğiyle beslenen kadim bir toprak parçasıdır Erzurum.



HTML5 ile menüler
<nav> etiketi içerisinde
kullanılmaya
başlanmıştır.

<nav> etiketi, sayfa içinde dolaşma (navigation) menüsünün tanımı için kullanılır. Aşağıdaki örnekte bir web sayfasında olabilecek menü linkleri *<nav>* etiketinin içinde verilmiştir.

Tablo 2.8. Nav Etiketi

HTML5 Etiketleri
<pre><!DOCTYPE html> <html> <head> <meta charset='UTF-8'> <title>ilk sayfam</title> </head> <body> <nav> <h2>MENU</h2> </pre>

```
<li><a href="anasayfa.html">ANA SAYFA</a></li>
<li><a href="hakkimda.html">HAKKIMDA</a></li>
<li><a href="projeler.html">PROJELERİM</a></li>
</ul>
</nav>
</body>
</html>
```

Web Sayfası Görüntüsü

MENU

- [ANA SAYFA](#)
- [HAKKIMDA](#)
- [PROJELERİM](#)

<section> etiketi sayfaların içindeki bölümleri belirtir. Web sayfasının çeşitli kısımlarını birbirinden ayırmak için kullanılabilir. Aşağıdaki örnekte HTML ve CSS bölümleri iki *<section>* etiketi içinde verilmiştir.

Tablo 2.9. Section Etiketi

HTML5 Etiketleri

```
<!DOCTYPE html>
<html>
<head>
<meta charset='UTF-8'>
<title>ilk sayfam</title>
</head>
<body>
<section>
<h1>HTML</h1>
<p>HTML (Hyper Text Markup Language) internet üzerinde web sayfası oluşturmak için kullanılan bir betik dilidir.
</p>
</section>
<section>
<h1>CSS</h1>
<p> CSS (Cascading Style Sheet) web sayfalarındaki html etiketlerinin ekranda nasıl görüneceğini tanımlamak için kullanılan stil kodlarıdır. </p>
</section>
</body>
</html>
```

Web Sayfası Görüntüsü



HTML5 ile sayfaları ve içeriği bölgelere ayırmak için *<section>* ve *<article>* etiketleri kullanılmaktadır.

HTML

HTML (Hyper Text Markup Language) internet üzerinde web sayfası oluşturmak için kullanılan bir betik dilidir.

CSS

CSS (Cascading Style Sheet) web sayfalarındaki html etiketlerinin ekranda nasıl görüneceğini tanımlamak için kullanılan stil kodlarıdır.

<article> etiketi, web sayfası üzerine yerleştirilecek haber, makale vb. metinleri tanımlamak ve yerleştirmek için kullanılır. Ayrıca, forum mesajları, blog gönderileri, haber hikâyesi, yorumlar gibi metinsel ifadelerin yerleştirilmesinde yaygın olarak tercih edilebilir. Aşağıdaki örnekte Atatürk Üniversitesi ile ilgili bilgiler *<article>* etiketi içinde verilmiştir.

Tablo 2.10. Article Etiketi

HTML5 Etiketleri

```
<!DOCTYPE html>
<html>
<head>
<meta charset='UTF-8'>
<title>ilk sayfam</title>
</head>
<body>
<article>
    <h1>Atatürk Üniversitesi</h1>
    <p><br>Atatürk Üniversitesi'nin tarihçesi, Türkiye Cumhuriyeti'nin
    önemli projelerinden birinin gerçekleşme öyküsüdür.
    <br><br>Kuruluş tarihi, 7 Haziran 1957 olarak ifade edilir.</p>
</article>
</body>
</html>
```

Web Sayfası Görüntüsü

Atatürk Üniversitesi

Atatürk Üniversitesi'nin tarihçesi, Türkiye Cumhuriyeti'nin önemli projelerinden birinin gerçekleşme öyküsüdür.

Kuruluş tarihi, 7 Haziran 1957 olarak ifade edilir.



HTML5 ile içerikler ayrı olarak *<aside>* etiketiyle sayfada verilebilir.

<aside> etiketi, sayfa içeriğinden ayrı olarak yer alması istenilen içeriği yerlestirecek alanı tanımlamak için kullanılır. Aşağıdaki örnekte yine Atatürk Üniversitesi ile ilgili bilgi, *<aside>* etiketi içinde verilmiştir.

Tablo 2.11. Aside Etiketi

HTML5 Etiketleri
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>İlk HTML belgesi</title> </head> <body> <p>Atatürk Üniversitesi Türkiye'nin en güzel kampüslerinden birine sahiptir. 6.5 milyon m²'lik açık alana sahip olan kampüste, kapalı alan 1 milyon m²'dir.
</p> <aside> <h4>Atatürk Üniversitesi</h4> <p> 23 fakülte, 1 yüksekokul, 1 konservatuvar, 12 meslek yüksekokulu, 8 enstitü, 25 araştırma merkezi vardır.
</p> </aside> </body> </html></pre>
Web Sayfası Görüntüsü
<p>Atatürk Üniversitesi Türkiye'nin en güzel kampüslerinden birine sahiptir. 6.5 milyon m²'lik açık alana sahip olan kampüste, kapalı alan 1 milyon m²'dir.</p> <p>Atatürk Üniversitesi</p> <p>23 fakülte, 1 yüksekokul, 1 konservatuvar, 12 meslek yüksekokulu, 8 enstitü, 25 araştırma merkezi vardır.</p>



HTML5'te `<figure>` etiketi; resim, fotoğraf, diyagram, kod listeleri gibi kendi içeriğini anlatan bir yapı oluşturur.

<footer> etiketi sayfanın altbilgi kısmının tanımlanması için kullanılır.

Sayfanın en alttaki alanıdır. Bazı önemli bilgiler, yasal hatırlatmalar (telif haklarını, kullanım gizliliği gibi) buraya yerleştirilir. Aşağıdaki örnekte web sayfası bilgisi `<footer>` etiketi içinde verilmiştir

Tablo 2.12. Footer Etiketi

HTML5 Etiketleri
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>ilk HTML belgesi</title> </head> <body> <footer> web adresi:https://www.atauni.edu.tr
 </footer> </body> </html></pre>
Web Sayfası Görüntüsü
web adresi: https://www.atauni.edu.tr



HTML5 ile sayfalardaki
altbilgi bölümü
<footer> etiketi içinde
verilebilir.

Buraya kadar anlatılan temel anlamsal etiketlere ilaveten <figure>, <figcaption>, <details>, <summary>, <progress>, <mark>, <time> gibi çok sayıda yeni semantik-anlamsal etiket HTML5'e dâhil edilmiştir.

<figure> etiketi; resim, fotoğraf, diyagram, kod listeleri gibi kendi içeriğini anlatan bir yapı oluşturur. *<figcaption> etiketi* de <figure> etiketi içinde yer alarak, görsel ya da görseller hakkında ek bilgiler vermek için kullanılır. Aşağıdaki örnekte logo <figure> etiketi içinde verilmiştir.



Bireysel Etkinlik

- HTML5 ile gelen yeni semantik-anlamsal etiketleri kullanarak okuduğunuz bölümle ilgili içeriği Şekil 2.1.'deki gibi tasarlamaçak şekilde bir web sayfası hazırlayın.

Tablo 2.13. Figure ve Figcaption Etiketi

HTML5 Etiketleri
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>İlk HTML belgesi</title>
</head>
<body>
<figure>
<figcaption>

Atatürk Üniversitesi, daha fazla bilgi için web sayfasını ziyaret ediniz.
</figcaption>
</figure>
</body>
</html>
Web Sayfası Görüntüsü

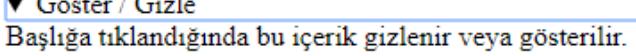
Atatürk Üniversitesi , daha fazla bilgi için web sayfasını ziyaret ediniz.



HTML5 ile <details> ve <summary> etiketleriyle bilgiler gösterilip/gizlenebilmektedir.

<details> ve <summary> etiketleri JavaScript dili kullanmadan bilgilerin gösterilip gizlenmesini sağlayan yeni etiketlerdir. <summary> etiketi <details> etiketinin başlığını tanımlar. Başlık bilgisi içeriğin gösterilip/gizlenmesi için tıklanabilmektedir. Aşağıdaki örnekte Göster/Gizle tıklandığında altındaki açıklama satırı gösterilir ya da gizlenir.

Tablo 2.14. Details ve Summary Etiketleri

HTML5 Etiketleri
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>İlk HTML belgesi</title>
</head>
<body>
<details >
<summary>Göster / Gizle</summary>
Başlığa tıklandığında bu içerik gizlenir veya gösterilir.
</details>
</body>
</html>
Web Sayfası Görüntüsü


<progress> etiketi görevin ilerleme durumunu tanımlar. Örneğin `<progress value="90" max="100"></progress>` şeklinde kullanıldığında yüzde 90'ı dolmuş bir gösterge çubuğu aşağıdaki gibi görünecektir. Value değeri değiştirilerek dolmuş olan bölüm değiştirilebilir.

Tablo 2.15. Progress Etiketi

HTML5 Etiketi ve Web Sayfası Görüntüsü
<code><progress value="90" max="100"></progress></code> 

<mark> etiketi işaretlenmiş ya da vurgulanmak istenilen metinleri tanımlar. Örneğin bir metnin bir bölümünün önemli olduğu düşünülüyorsa *<mark>* etiketi kullanılabilir.

Tablo 2.16. Mark Etiketi

HTML5 Etiketi ve Web Sayfası Görüntüsü
<code><p>Internet tabanlı programlama <mark>HTML5</mark> eğitimi</p></code> Internet tabanlı programlama HTML5 eğitimi

<time> etiketi tarih ve saat verilerini kapsar. 24 saatlik bir zaman dilimini ya da tarihi göstermek için kullanılır.

Tablo 2.17. Time Etiketi**HTML5 Etiketi ve Web Sayfası Görüntüsü**

```
<p>Yarın saat <time>12:00</time>'de görüşmem var.</p>
<p>Önümüzdeki <time datetime="2019-08-26">pazartesi</time> tatil yapacağız.</p>
```

Yarın saat 12:00'de görüşmem var.

Önümüzdeki pazartesi tatil yapacağız.

 HTML5 ile etiketler iç içe kullanılabilmektedir.

HTML5 dilinin bir özelliği de etiketler iç içe geçebilir veya bir etiket diğer etiketlerin alt etiketi şeklinde kullanılabilir. Aşağıdaki örnekte olduğu gibi bir article elementinin kendisine ait section, header gibi elementleri olabilir.

Tablo 2.18. İç içe etiketler**HTML5 Etiketleri**

```
<article>
  <section>
    <header>
      <h1>Atatürk Üniversitesi</h1>
    </header>
    <p>http://www.ktu.edu.tr/</p>
  </section>
  <section>
    <header>
      <h1>Karadeniz Teknik Üniversitesi</h1>
    </header>
    <p>https://www.atauni.edu.tr</p>
  </section>
</article>
```

Web Sayfası Görüntüsü

Atatürk Üniversitesi

<http://www.ktu.edu.tr/>

Karadeniz Teknik Üniversitesi

<https://www.atauni.edu.tr>

HTML5 ile Yeni Form Elementleri

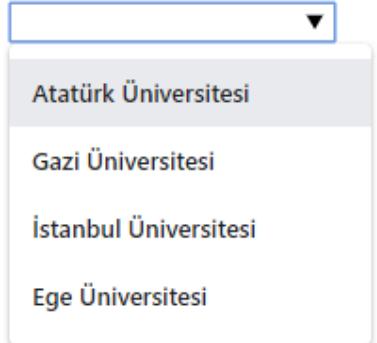
HTML5 ile yeni form etiketleri dâhil olmuştur. Bu yeni HTML etiketleriyle form elemanları çeşitlilik göstermektedir. Aşağıda yeni form elemanlarının bazıları verilmiştir:

<datalist> elementi, bir **<input>** elementi ile beraber kullanılarak önceden belirlenmiş bir dizi seçenekin listesini oluşturur. Hazır gelen bu liste içinden seçim yapılabilir.



HTML5 ile datalist, keygen, output gibi yeni form elementleri gelmiştir.

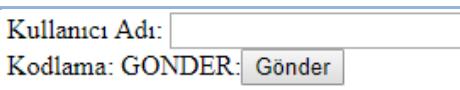
Tablo 2.19. Datalist Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<pre><input list="universiteler"> <datalist id="universiteler"> <option value="Atatürk Üniversitesi"> <option value="Gazi Üniversitesi"> <option value="İstanbul Üniversitesi"> <option value="Ege Üniversitesi"> </datalist></pre>	

Yukarıdaki örnekte görüldüğü üzere, `<datalist>` elemanı bilgi girişinde kullanıcıya kolaylık sağlamak amacıyla bir açılır menü içinde üniversite isimleri listelemiştir. Ayrıca bir harf girilirse, menüdeki sadece o harf ile ilişkili seçenekler listelenecektir. Örneğin G harfine basıldığından sadece Gazi Üniversitesi seçeneği görülecektir.

<keygen>, HTML5'te bu yeni form etiketi ile kimlik denetimi için güvenli bir yöntem oluşturulur. Form verisi gönderildiği zaman, biri özel biri genel iki anahtar üretilir. Özel anahtar sizin yerel bilgisayarınızda saklanır, genel olanı ise sunucuya gönderilir.

Tablo 2.20. Keygen Etiketi

HTML Etiketleri	Web Sayfası Görüntüsü
<pre><form action="sifreleme.php" method="post"> Kullanıcı Adı: <input type="text" name="ad"> Kodlama: <keygen name="sifrele">
GÖNDER:<input type="submit" name="gonder" > </form></pre>	

<output>, HTML5'te bu etiket ile hesaplamaların sonucu çıktı olarak görülebilir. Aşağıda range ve number tipinde iki adet girdi alanı örneği verilmiştir. Bu girdi alanlarından gelen veriler toplama işlemi yapıldıktan sonra sonucu yazdırılmaktadır.



HTML5 ile Adobe Flash
ve Applet gibi
teknolojilerin yakın
zamanda tarihe
karışacağı tahmin
ediliyor.

Tablo 2.21. Output Etiketi

HTML5 Etiketi
<form oninput= "ab.value= parseInt(a.value)+parseInt(b.value)"> 0<input type="range" id="a" value="500">100 +<input type="number" id="b" value="500"> =<output name="ab" for="a b"></output></form>
Web Sayfası Görüntüsü


Yukarıdaki örnekte, 0-100 arasındaki değerler fare yardımıyla hareket ettirildiğinde değişecektir. Sonuç otomatik olarak sağda verilecektir.

Özet olarak, HTML5 birçok yeniliği beraberinde getirmiştir. Görünen o ki her yeni gelen sürümle birlikte web sayfalarında değişiklikler olacaktır. Yaygın olarak kullanılan bazı teknolojiler saf dışı olacak, yeni özellikler hayatımıza hızlıca girecek. Örneğin web sayfalarında animasyon ve oyun gibi etkileşimler geliştirmek için sıkılıkla kullanılan Adobe Flash programının kullanımı azalacak ya da hiç kullanılmasına gerek kalmayacaktır. Java tabanlı geliştirilen Applet'ler yakın zamanda tarihe karışacak. Kitabın bu bölümü itibarıyla HTML5'in getirdiği yenilikleri görmeye başladık, sonraki iki bölümde de HTML5'in yeni etiketlerini ve kullanım özelliklerini inceleyeceğiz.



:Özet

- HTML5, HTML dilinin en son sürümüdür. Bu yeni sürüm ile HTML dili yeni stratejiler ve hedefler ortaya koyan ve farklı web tarayıcıları için standartizasyonu amaçlayan yeni nesil bir teknoloji olarak görülmektedir.
- HTML5, 2014 yılında W3C (World Wide Web Consortium) desteğiyle daha önceki sürümlerin birçok özelliğini barındırmakla beraber yeni oluşturulan ve güncel ihtiyaçlara cevap verebilecek teknoloji olarak piyasaya sürülmüştür.
- HTML 5'İN TEMEL ÖZELLİKLERİ**
- HTML5, önceki sürümlerle uyumlu olmakla beraber sadeleştirilmiş ve düzeltilmiştir. Gelişmiş çoklu ortam desteği sunmaktadır.
- HTML5, üçüncü parti yazılımlara ve eklentilere ihtiyaç duymadan çoklu ortam ve grafik görüntüleri oynatabilir.
- HTML5, kompleks web uygulamaları için, konum belirleme, sürükle bırak, yerel depolama gibi yeni API'ler sunar.
- JavaScript ve CSS teknolojilerinin daha etkili bir şekilde kullanımına imkân tanır.
- HTML5 ile Değişen Etiketler**
- HTML5, HTML'in eski sürümlerinden bağımsız bir sürüm olmayıp daha önceden kullanılan standartlar hâlâ devamlılığını sürdürmektedir.
- HTML5 ile ilgili ilk yenilik DOCTYPE alanına gelen değişiklikle olmuştur. Kullanımı şöyledir: <!DOCTYPE html>
- Değişikliğe gidilen bir diğer etiket ise META etiketidir. Kullanımı şöyledir: <meta charset="UTF-8">
- HTML5'te style ve script etiketlerinin kullanımında da değişikliğe gidilmiştir. Kullanımı şöyledir: <style> css kodları </style>
- HTML5 ile Yeni Semantik-Anlamsal Etiketler**
- <article> içeriğin makale, haber olduğunu belirler.
- <aside> içeriğin ana içerik yanında harici içerik olduğunu belirler.
- <details> özeti için detay bilgi tanımlar.
- <figcaption> <figure> etiketiyle beraber kullanılır, başlık belirtir.
- <figure> içeriğin resim, şekil gibi görsel olduğu belirtir.
- <footer> sayfanın alt bölüm bilgisi olduğunu belirtir.
- <header> sayfanın üst bölüm bilgisi olduğunu belirtir.
- <mark> içeriğin vurgulanmış olduğunu belirtir.
- <nav> içeriğin menü değeri olduğunu belirtir.
- <progress> görev ilerleme çubuğu tanımlar.
- <section> içeriğin sayfanın bir bölümü olduğunu belirtir.
- <summary> <details> etiketiyle kullanılır, başlık tanımlar.
- <time> içeriğin tarih/saat olduğunu belirtir.
- HTML5 ile Yeni Form Elementleri**
- <datalist> veri girişi için önceden tanımlanmış seçeneklerin bir listesini belirtir. Kullanıcılar, veri girdiklerinde önceden tanımlanmış seçeneklerin açılır listelerini görürler.
- <keygen> kullanıcıların kimliğini doğrulamak için güvenli bir yol sağlamaktır. Form gönderildiğinde, biri özel, biri de genel olmak üzere iki anahtar oluşturulur. Özel anahtar yerel olarak saklanır ve genel anahtar sunucuya gönderilir.
- <output> hesaplama veya komut sonucunu temsil eder.
- HTML5 teknolojilerinin kullanımının web sayfalarında yaygınlaşlığı görülmektedir. Google başta olmak üzere Facebook, Twitter, LinkedIn, Apple gibi büyük yazılım firmaları tercih etmektedir.

DEĞERLENDİRME SORULARI

1. Aşağıdakilerden hangisi HTML dilinin en son sürümüdür?
 - a) HTML 4.01
 - b) HTML 5
 - c) HTML 3.8
 - d) HTML 6
 - e) HTML 7
2. HTML5'te bulunan `<section>` etiketi önceki HTML sürümlerinde aşağıdaki etiketlerden hangisine karşılık gelmektedir?
 - a) `<p align="left">`
 - b) `<div id="content">`
 - c) `<body bgcolor="red">`
 - d) `<container>`
 - e) `<head>`
3. Sitenizin menülerinin bulunacağı kısmı HTML5'te aşağıdaki etiketlerden hangisi ile tanımlanmaktadır?
 - a) `<nav>`
 - b) `<menü>`
 - c) `<footer>`
 - d) `<figcaption>`
 - e) `<summary>`
4. web sayfası üzerine yerleştirilecek haber, makale vb. metinleri tanımlamak ve yerleştirmek için kullanılır.
Cümlede boş bırakılan yere aşağıdakilerden hangisi getirilmelidir?
 - a) `<section>`
 - b) `<article>`
 - c) `<head>`
 - d) `<nav>`
 - e) `<footer>`
5. HTML dilinde DOCTYPE alanı şu şekilde tanımlanıyordu:
`<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">`
Aşağıdakilerden hangisi DOCTYPE alanının HTML5 ile gelen değişiklerden sonraki hâlidir?
 - a) `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN">`
 - b) `<!DOCTYPE html "http://www.w3.org/xhtml1-transitional.dtd">`
 - c) `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict">`
 - d) `<!DOCTYPE html PUBLIC "-//W3C`
 - e) `<!DOCTYPE html>`

6. Aşağıdakilerden hangisi HTML5'in gelişimine destek veren gruplardandır?
 - a) W3C (World Wide Web Consortium)
 - b) CSS
 - c) Google
 - d) Web
 - e) İnternet
7. Aşağıdakilerden hangisi HTML5 ile değişen etiketlerden biridir?
 - a)

 - b) <p>
 - c) <table>
 - d) <meta>
 - e) <body>
8. Web sayfamızdaki bazı bilgileri gösterip/gizleme yapmak istiyorsak hangi HTML5 etiketini kullanırız?
 - a) <mark>
 - b) <figure> ve <figcaption>
 - c) <details> ve <summary>
 - d) <progres>
 - e) <section> ve <aside>
9. Aşağıdaki form etiketlerinden hangisiyle görseldekine benzer hesaplamaların sonucu çıktı olarak görülebilir?



- a) datalist
 - b) keygen
 - c) output
 - d) mark
 - e) time
10. HTML5'te aşağıdaki form etiketlerinden hangisiyle kimlik denetimi için güvenli bir yöntem oluşturulur?
 - a) datalist
 - b) mark
 - c) progres
 - d) keygen
 - e) aside

Cevap Anahtarı

1.b, 2.b, 3.a, 4.b, 5.e, 6.a, 7.d, 8.c, 9.c, 10.d

YARARLANILAN KAYNAKLAR

HTML5 & CSS. 6 Ağustos 2019 tarihinde

<https://gelecegiyazanlar.turkcell.com.tr/konu/web-programlama/egitim/201> adresinden erişildi.

HTML5 Dersleri. 6 Ağustos 2019 tarihinde <https://www.yusufsezer.com.tr/html5-dersleri/> adresinden erişildi.

HTML5 Dersleri. 6 Ağustos 2019 tarihinde https://www.w3schools.com/html/html5_intro.asp adresinden erişildi.

HTML5. 6 Ağustos 2019 tarihinde <http://www.w3.org/TR/html5> adresinden erişildi.

HTML5 İLE YENİ VERİ GİRİŞİ VE FORM ELEMANLARI



- HTML5 ile Yeni Veri Giriş (Input) Tipleri
- HTML5 ile Yeni Form Özellikleri
- Form Örnekleri

Atatürk Üniversitesi
Açıköğretim Fakültesi



İNTERNET
PROGRAMCILIĞI II
Doç. Dr. İbrahim
ÇETİN

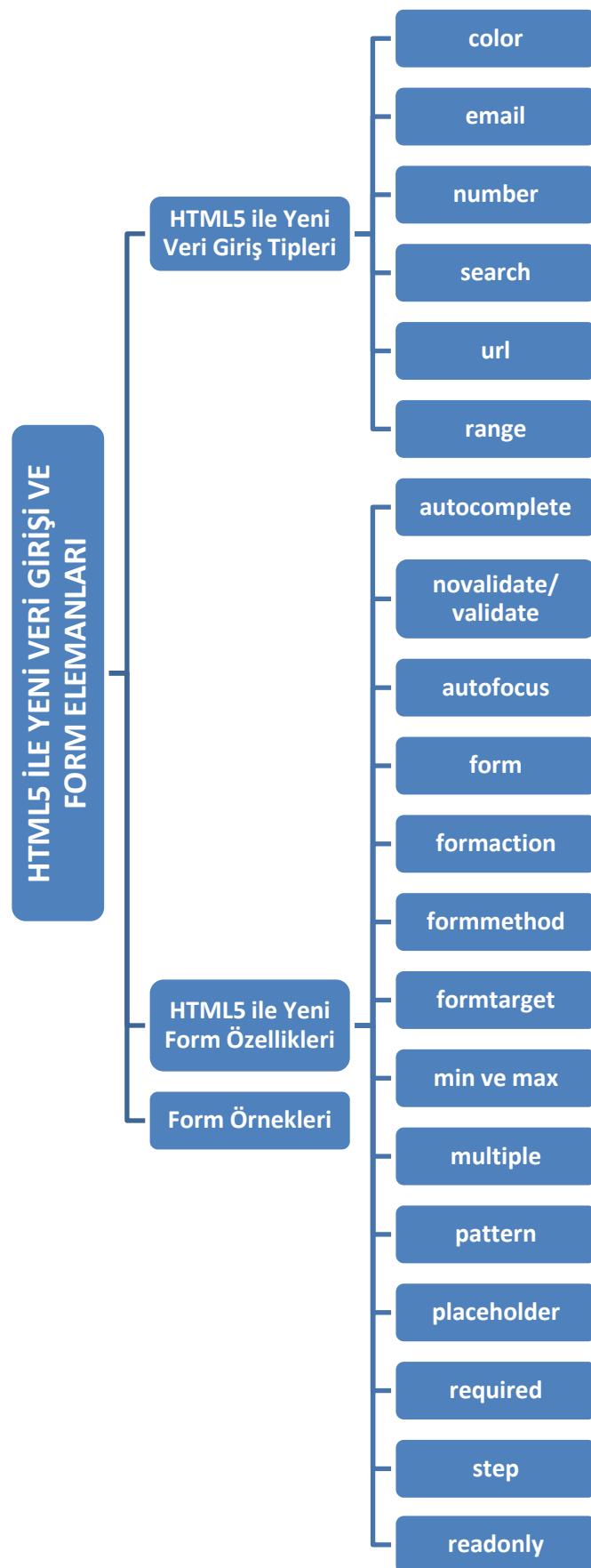
İÇİNDEKİLER



- Bu üniteyi çalıştırınca sonra;
- HTML5'in yeni giriş tiplerini sıralayabilecek,
- HTML5'in yeni form özelliklerini kullanabilecek,
- Yeni giriş tipleriyle form özelliklerini birleştirebilecek,
- Etkileşimli formlar oluşturabileceksiniz.

HEDİFLER

ÜNİTE
3



GİRİŞ

Bu bölümde HTML5'in giriş () elemanlarından bahsedilecektir. Önceki haftalardan hatırlayacağınız üzere <input> etiketi, <form> ve </form> etiketleri arasında yer alır ve kullanıcılarından farklı tür veri girişi yapılan alanlar tanımlamaktadır. Hatta veri girişlerinden bahsedildiğinde akla sadece birkaç tip gelirdi (button, submit vb.). HTML5 ile beraber veri girişi tiplerinde çeşitliliğe gidilmiştir. Yeni veri girişi tiplerinden bazıları şunlardır: color, email, number, search, url, range ve tarih. Yeni veri girişi etiketleriyle birkaç satır html koduyla hem görsel hem de kontrollerin yapılacak veri girişleri mümkün hâle gelmiştir. Çünkü bu yeni veri girişi etiketleri farklı parametreler de alabilmektedir. Bu parametrelerden bazıları şöyle sıralanabilir: autocomplete, novalidate/validate, autofocus, formaction, formnovalidate, formtarget, min ve max, multiple, pattern (regexp), placeholder, required, step, readonly. Bu yeni gelen özelliklerle beraber çok daha etkileşimli formlar yapılmaktadır.



HTML5 ile <input> etiketiyle kullanılan çok sayıda yeni veri giriş tipi ortaya çıkmıştır.

Bu bölümde öncelikle örneklerle yeni veri girişi tiplerinden bahsedilecektir. Her bir veri tipiyle uygulamalar yapılacaktır. Daha sonra bu veri tipleriyle yaygın olarak kullanılan parametreler anlatılacaktır. Örneğin veri giriş alanlarının otomatik olarak doldurulup doldurulmayacağından başlanacak, dolduruldu ise bazı kriterlere göre doğru doldurulduğunun kontrol edilmesine kadar çok sayıda uygulama yapılacaktır. Son olarak farklı form tasarımları yapılarak birden fazla girdi özelliğinin beraber kullanımı gösterilecektir. Kisaca bu bölümün sonunda HTML5 ile beraber gelen yeni veri girişi etiketleri kullanarak kendi formlarınızı tasarlayarak etkileşimli sayfalar yapabileceksiniz.

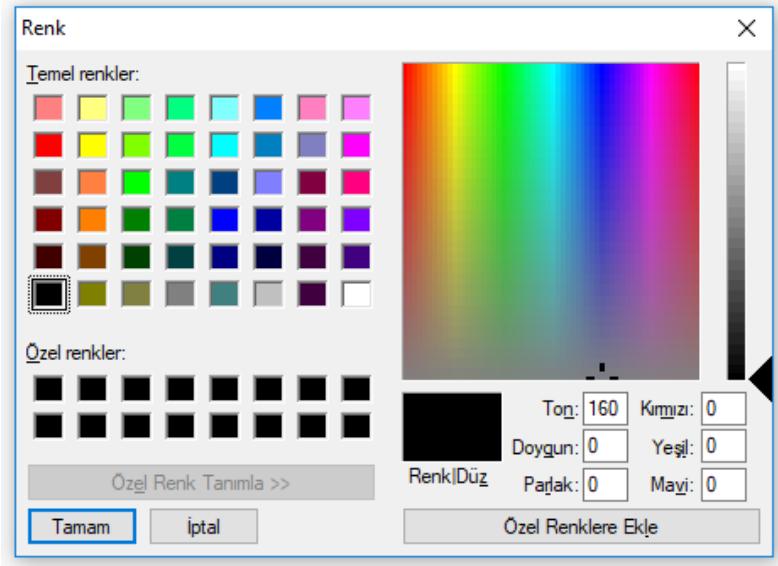
HTML5 İLE YENİ VERİ GİRİŞ (INPUT) TİPLERİ

HTML5 ile veri girişi tipleri çeşitlilik göstermektedir. Aşağıda yeni veri girişi tiplerinden öne çıkanlar sıralanmıştır, sonrasında ise her biriyle ilgili örnekler verilmiştir:

- color
- email
- number
- search
- url
- range
- ve tarih ile ilgili veri giriş tipleri (date, datetime, time vb.)

Color yeni veri tipi <input> alanı için renk tanımlamakta kullanılır. Bu veri tipiyle sayfada renk kutucuğu gösterilir. Bu kutucuk seçildiğinde ise seçim yapılmabilmesi için yeni renk penceresi açılır. Aşağıdaki örnekte görüldüğü üzere favori rengin seçilmesi istenilmiştir ve seçim için yeni renk penceresinde açılmıştır.

Tablo 3.1. Color Veri Tipi

HTML5 Veri Tipi
<pre><form> Favori renginizi seçin: <input type="color" name="renksecimi">
 </form></pre>
Web Sayfası Görüntüsü
<p>Favori renginizi seçin: </p> 



Number veri tipi alt ve üst limitleriyle kullanılır.

Email veri tipi e-posta alanı olması gereken giriş alanları için kullanılır. Tarayıcı destegine göre e-posta giriş kontrolü de yapılır. Aşağıdaki tabloda e-posta girişi örnek olarak verilmiştir.

Tablo 3.2. Email Veri Tipi

HTML5 Veri Tipi
<pre><form> E-mail: <input type="email" name="eposta">
 </form></pre>
Web Sayfası Görüntüsü
<p>E-mail: <input type="text" value="deneme@gmail.com"/></p>

Number veri tipi sadece sayısal veri girişleri için kullanılır. Alt ve üst (min, max) limitler verilebilir. Tarayıcı destegine göre giriş kontrolü de yapılır. Aşağıdaki örnekte 1-10 arasında sayı girilmesi beklenmektedir.

Tablo 3.3. Number Veri Tipi

HTML5 Veri Tipi
<form> Miktar girin (1 ve 10 arasında): <input type="number" name="miktar" min="1" max="10"> </form>
Web Sayfası Görüntüsü
Miktar girin (1 ve 10 arasında): <input type="number" value="5"/>

Search veri tipi ise arama alanları için kullanılır. Yine input etiketi kullanılarak type özelliği search yapılip ekranada bir metin kutusu oluşturulur. Aşağıdaki örnekte basit bir arama girişi gösterilmektedir.

Tablo 3.4. Search Veri Tipi

HTML5 Veri Tipi
<form> Google'da ara: <input type="search" name="googleara"> </form>
Web Sayfası Görüntüsü
Google'da ara: <input type="search"/>

Url veri tipi giriş alanı bir URL bağlantısı içerecekse kullanılır. Yine formlar doldurulurken istenileBILECEK bu veri tipi input etiketi kullanılarak type özelliği url yapılip ekranada bir metin kutusu oluşturulur. Aşağıdaki örnekte basit bir url girişi gösterilmektedir.

Tablo 3.5. Url Veri Tipi

HTML5 Veri Tipi
<form> Ana sayfanızı ekleyin: <input type="url" name="homepage"> </form>
Web Sayfası Görüntüsü
Ana sayfanızı ekleyin: <input type="url" value="www.atauni.edu.tr"/>

Range veri tipi giriş alanı belirli bir aralıkta olan bir sayının seçilmesi gerekiyorsa kullanılır. Aslında number parametresine benzer şekilde kullanıldığı söylenebilir. Aşağıdaki örnekte basit bir range seçimi gösterilmektedir.

Tablo 3.6. Range Veri Tipi

HTML Veri Tipi
<form> <input type="range" name="points" min="1" max="10"> </form>
Web Sayfası Görüntüsü


HTML5 ile *tarih kavramıyla ilgili de* çok sayıda yeni veri tipi çeşitleri eklenmiştir. Bunlardan bazıları şöyle sıralanabilir:

- **date:** Kullanıcıya bir takvim içinden zaman işaretleme olanağı sağlar.
- **datetime:** Kullanıcıya tarih ve zaman seçme olanağı sağlar.
- **time:** Yukarı/ aşağı oklarıyla zaman bilgisi girmek için kullanılır.
- **month:** Kullanıcıya yıl ve ay seçme olanağı sağlar.
- **week:** Kullanıcıya bir yıl ve hafta seçme olanağı sağlar.

Aşağıdaki tabloda bu yeni veri tiplerinin kullanımları ve web sayfasındaki görüntüleri verilmiştir.



Tarih ile ilgili çok sayıda veri tipi vardır: date, datetime, time, month, week sadece birkaçıdır.

Tablo 3.7. Tarih Veri Tipi

HTML5 Veri Tipi																																										
<form> Doğum günü (tarih ve saat): <input type="datetime" name="dgunutarihsaat"> Bir saat seçin: <input type="time" name="saat"> Bir hafta seçin: <input type="week" name="hafta"> Doğum günü (ay ve yıl): <input type="month" name="dgunuay"> Doğum günü: <input type="date" name="dogumgunu"> </form>																																										
Web Sayfası Görüntüsü																																										
<p>Doğum günü (tarih ve saat): <input type="text" value="01.12.1999"/></p> <p>Bir saat seçin: <input type="text" value="23 : 59"/></p> <p>Bir hafta seçin: <input type="text" value="32 . hafta, 2019"/></p> <p>Doğum günü (ay ve yıl): <input type="text" value="Ağustos 2019"/></p> <p>Doğum günü: <input type="text" value="05 . 08 . 2019"/> x ▼</p> <div style="border: 1px solid black; padding: 5px; width: fit-content;"> <p style="margin: 0;">Ağustos 2019 ▾</p> <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th>Pzt</th><th>Sal</th><th>Çar</th><th>Per</th><th>Cum</th><th>Cmt</th><th>Paz</th></tr> </thead> <tbody> <tr> <td>29</td><td>30</td><td>31</td><td>1</td><td>2</td><td>3</td><td>4</td></tr> <tr> <td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td></tr> <tr> <td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td></tr> <tr> <td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td></tr> <tr> <td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td><td>1</td></tr> </tbody> </table> </div>	Pzt	Sal	Çar	Per	Cum	Cmt	Paz	29	30	31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1
Pzt	Sal	Çar	Per	Cum	Cmt	Paz																																				
29	30	31	1	2	3	4																																				
5	6	7	8	9	10	11																																				
12	13	14	15	16	17	18																																				
19	20	21	22	23	24	25																																				
26	27	28	29	30	31	1																																				



Opera ve Chrome tarayıcıları yeni veri tiplerinin hepsini desteklemektedir.

Örnek

- Birkaç örnek:
- <input type="color" name="renk">
- <input type="date" name="tarih">
- <input type="email" name="email">
- <input type="search" name="ara">

HTML5 ile gelen bu yeni girdi (input) etiketleri birçok tarayıcı tarafından desteklenmektedir. Bazı tarayıcılar henüz güncelleme yapmamasına rağmen yakın zamanda hepsi tarafından desteklenmesi beklenmektedir. Aşağıda hangi tarayıcının hangi sürümle beraber desteklemeye başladığı görülmektedir.

Tablo 3.8. Tarayıcılar Hakkında

Veri Tipleri	IE	Firefox	Opera	Chrome	Safari
color	Hayır	Hayır	11.0	12	Hayır
url	Hayır	4.0	9.0	10.0	Hayır
number	Hayır	Hayır	9.0	7.0	Hayır
range	Hayır	Hayır	9.0	4.0	4.0
email	Hayır	4.0	9.0	10.0	Hayır
Search	Hayır	4.0	11.0	10.0	Hayır
Tarih (date, time vb.)	Hayır	Hayır	9.0	10.0	Hayır

HTML5 İLE YENİ FORM ÖZELLİKLERİ



HTML5 ile çok sayıda yeni form özelliği kullanılmaya başlanmıştır.

HTML5 ile form elemanlarına yeni özellikler kazandırılmıştır. Özellikle input etiketiyle beraber kullanılan bu özellikler formların kullanışılığını artırmıştır. Aşağıda yeni eklenen özellikler verilmektedir:

- autocomplete
- novalidate/validate
- autofocus
- form
- formaction
- formmethod
- formnovalidate
- formtarget
- min ve max
- multiple
- pattern (regexp)
- placeholder
- required
- step
- readonly

Autocomplete özelliği, bir form ya da girdi alanının on/off değerleri alarak otomatik olarak doldurulup doldurulmayacağı belirler. Eğer autocomplete açıksa (on durumda ise), tarayıcı kullanıcının önceden belirlediği değerleri kullanarak ilgili alanları otomatik olarak doldurur. autocomplete özelliği form elemanlarından text, search, url, tel, email, password, datepickers, range ve color gibi birçok girdi çeşidiyle kullanılabilir.

Tablo 3.9. Autocomple Özelliği

HTML5 Form Özelliği
<pre><form autocomplete="on"> AD: <input type="text" name="ad">
 SOYAD: <input type="text" name="soyad">
 E-mail:<input type="email" name="email"
 <input type="submit"> </form></pre>
Web Sayfası Görüntüsü
<p>The screenshot shows a web form with an input field for 'E-mail'. The user has typed 'Şehriban' into the field. A dropdown menu is open, displaying several suggestions: 'ASLIHAN', 'Firdevs', 'ezgi', and 'ASENA'. To the right of the input field is a 'Gönder' (Send) button.</p>

Novalidate ya da validate özelliği mantıksal (true/false) bir özelliktir. Eğer novalidate kullanılırsa form verisinin doğrulanmasına gerek yoktur eğer validate kullanılırsa doğruluk/geçerlilik kontrolü yapılacaktır. Aşağıdaki örnekte E-mail olarak girilen abc değerleri kontrol edilip içerisinde @ işaretinin olmadığı kullanıcıya uyarı olarak bildirilmektedir.

Tablo 3.10. Validate Özelliği

HTML5 Form Özelliği
<pre><form validate> E-mail: <input type="email" name="eposta"> <input type="submit"> </form></pre>
Web Sayfası Görüntüsü
<p>The screenshot shows a web form with an input field for 'E-mail'. The user has typed 'abc' into the field. A validation error message is displayed in a box: 'Lütfen e-posta adresine bir "@" işaretini ekleyin. "abc" adresinde "@" eksik.'</p>

Autofocus özelliği de mantıksal (true/false) bir özelliktir. Kullanılırsa sayfa yüklenliğinde ilgili <input> elemanına veri girişi yapmak üzere odaklanır (focus). Aşağıdaki örnekte sayfa yüklenliğinde imleç AD veri girişine gitmektedir.



Mantıksal ifadeler
True/False
(doğru/yanlış) ile
kullanılır.

Tablo 3.11. Autofocus Form Özelliği

HTML5 Form Özelliği
<form> AD: <input type="text" name="ad" autofocus> SOYAD: <input type="text" name="soyad"> <input type="submit"> </form>
Web Sayfası Görüntüsü
AD: <input name="ad" style="border: 1px solid blue; width: 150px; height: 20px;" type="text" value=""/> SOYAD: <input name="soyad" style="width: 150px; height: 20px;" type="text" value=""/> <input type="button" value="Gönder"/>

Form özelliği ile form elemanlarının form dışında da tanımlanmasına izin verilmektedir. Böylelikle form elemanları farklı formlarda da tanımlanabilir.

Tablo 3.12. Form Özelliği

HTML5 Form Özelliği
<form id="form1"> AD: <input type="text" name="ad"> <input type="submit" value="GONDER"> </form>
<p>Soyad formun dışında olmasına rağmen form özelliği ile form1 elemanı olarak görülmektedir.</p> SOYAD: <input type="text" name="soyad" form="form1">

Formaction özelliğe; form gönderildiği zaman, girdi kontrolü ile farklı bir dosyaya gönderim imkânı sağlanabilir.

Tablo 3.13. Formaction Özelliği

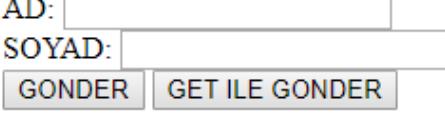
HTML5 Form Özelliği
<<form action="demo.php" > AD: <input type="text" name="ad"> SOYAD: <input type="text" name="soyad"> <input type="submit" value="GONDER"> <input type="submit" formaction="kontrol.php" value="Yöneticiye Gönder"> </form>
Web Sayfası Görüntüsü
AD: <input name="ad" style="border: 1px solid blue; width: 150px; height: 20px;" type="text" value=""/> SOYAD: <input name="soyad" style="width: 150px; height: 20px;" type="text" value=""/> <input type="button" value="GONDER"/> <input type="button" value="Yöneticiye Gönder"/>



Formnovalidate veya Formvalidate ile form elemanları kontrol edilebilmektedir.

Formmethod özelliği, form verisini farklı bir adrese farklı metot ile gönderme imkânı sunar. Aşağıdaki örnekte GET İLE GÖNDER girdi ile form verileri farklı bir URL'ye gönderilmiş olacaktır.

Tablo 3.14. Formmethod Özelliği

HTML5 Form Özelliği
<form method="post"> AD: <input type="text" name="ad"> SOYAD: <input type="text" name="sad"> <input type="submit" value="GÖNDER"> <input type="submit" formmethod="get" formaction="kontrol.php" value="GET İLE GÖNDER"> </form>
Web Sayfası Görüntüsü


Formnovalidate özelliği de mantıksal (true/false) bir özelliktir. Kullanılırsa form elemanları kontrol edilmeden gönderilecektir. Aşağıdaki örnekte e-posta girdisi herhangi bir kontrol olmadan gönderilmiştir.

Tablo 3.15. Formnovalidate Özelliği

HTML5 Form Özelliği
<form action="kontrol.php"> E-posta: <input type="email" name="eposta"> <input type="submit" formnovalidate value="Gönder"> </form>

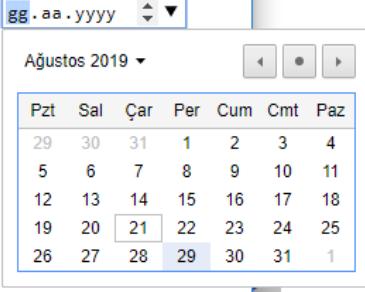
Formtarget özelliği, form verileri gönderildikten sonra nerede görüntüleneceğini gösteren bir isim ya da anahtar sözcük belirler. Aşağıdaki örnekte form verileri gönderildikten sonra formtarget ile yeni pencere açılmıştır.

Tablo 3.16. Formtarget Özelliği

HTML5 Form Özelliği	
<form action="kontrol.php">	
AD: <input type="text" name="ad"> 	
SOYAD: <input type="text" name="soyad"> 	
<input type="submit" value="GÖNDER">	
<input type="submit" formtarget="_blank" value="YENİ PENCERE AÇARAK-GÖNDER">	
</form>	
Web Sayfası Görüntüsü	
<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> AD: <input type="text"/> </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> SOYAD: <input type="text"/> </div> <div style="background-color: #e0e0e0; border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> GÖNDER <input type="submit" value="YENİ PENCERE AÇARAK-GÖNDER"/> </div>	

Min ve max özellikleri number, range, date, datetime, datetime-local, month, time ve week gibi girdilerin en küçük ve en büyük değerlerini belirlenmesini sağlar.

Tablo 3.17. Min/Max Özelliği

HTML5 Form Özelliği	
<form >	
2020-01-01 DEN ÖNCEKİ BİR TARİH GİRİN:	
<input type="date" name="gun" max="2019-12-31"> 	
ADET GİRİN (1-10):	
<input type="number" name="miktar" min="1" max="10"> 	
<input type="submit">	
</form>	
Web Sayfası Görüntüsü	
2020-01-01 DEN ÖNCEKİ BİR TARİH GİRİN: <input type="date"/> ADET GİRİN (1-10): <input type="number" value="5"/> <input type="button" value="Gönder"/>	
	

Multiple özelliği, e-posta ve dosya seçimi gibi form elemanlarına birden fazla değer girilmesini sağlar. Aşağıdaki örnekte birden fazla dosya (file) seçim yapılabilme örneği verilmiştir.

 Multiple özelliği, e-posta ve dosya seçimi gibi form elemanlarına birden fazla değer girilmesini sağlar.

Tablo 3.18. Multiple Özelliği

HTML5 Form Özelliği
<form> RESİM SEÇ: <input type="file" name="resim" multiple> <input type="submit"> </form>
Web Sayfası Görüntüsü
RESİM SEÇ: <input type="button" value="Dosyaları Seç"/> Dosya seçilmedi <input type="button" value="Gönder"/>

Pattern özelliği ile; text, search, url, tel, email ve password gibi veri girişlerinin kontrollü yapılması sağlanır. Aşağıdaki örnekte dil kodu girilirken 2 harfli ve sadece metin değerleri girilebilmektedir. Aksi hâlde uyarı mesajı verilmektedir.

Tablo 3.19. Pattern Özelliği

HTML5 Form Özelliği
<form> DİL KODU: <input type="text" name="u_kod" pattern="[A-Za-z]{2}" title="İKİ HARFLIK KOD"> <input type="submit"> </form>
Web Sayfası Görüntüsü
DİL KODU: <input type="text" value="ENG"/> <input type="button" value="Gönder"/> <div style="border: 1px solid #ccc; padding: 5px; width: fit-content;"> ! Lütfen istenen biçimini eşleştirin. İKİ HARFLİK KOD </div>

Placeholder özelliği ile bir giriş alanına beklenen değerini açıklayan kısa bilgi verilebilir. Aşağıdaki örnekte AD girilmesi beklenen veri girişi kutusunda isminizi giriniz şeklinde kısa bilgi verilmiştir.

Tablo 3.20. Placeholder Özelliği

HTML5 Form Özelliği
<form> <input type="placeholder" name="AD" placeholder="İsminizi giriniz"> </form>
Web Sayfası Görüntüsü
<input type="text" value="İsminizi giriniz"/>

Required özelliği, form gönderilmeden önce istenen girdilerin doldurulmasını zorunlu kılar. Aşağıdaki örnekte, ad girişi yapılmadan göndere basıldığında uyarı mesajı alınmaktadır.

Tablo 3.21. Formnovalidate Özelliği

HTML5 Form Özelliği
<form> KULLANICI ADI: <input type="text" name="ad" required> <input type="submit"> </form>
Web Sayfası Görüntüsü
KULLANICI ADI: <input type="text"/> <div style="border: 1px solid #ccc; padding: 5px; width: fit-content;"> Lütfen bu alanı doldurun. </div>

Step özelliği, genellikle min ve max özellikleriyle birlikte kullanılarak geçerli sayı aralığında veri girişi yapılmasını sağlar. Örneğin, step=3 olarak belirtilmişse, ...-6,-3,0,3 gibi sayılar geçerli sayılar olacaktır.

Tablo 3.22. Step Özelliği

HTML5 Form Özelliği
<form> <input type="number" name="veri" step="3" min="-30" max="30"> <input type="submit"> </form>
Web Sayfası Görüntüsü
9 <input type="button" value="Gönder"/>

Readonly özelliği ile sadece okumaya izin verilir, değişikliğe veya yeni veri girişine izin verilmez. Aşağıdaki örnekte de görüldüğü üzere, ülke verisi Türkiye olarak girilmiş ve değiştirilmemektedir.

Tablo 3.23. Readonly Özelliği

HTML5 Form Özelliği
<form> Ülke: <input type="text" name="ulke" value="Türkiye" readonly> <input type="submit"> </form>
Web Sayfası Görüntüsü
Ülke: <input type="text" value="Türkiye"/> <input type="button" value="Gönder"/>

Form Örnekleri

Buraya kadar gösterilen veri tipi girişlerinden bazılarını içeren bir örnek form hazırlayalım. Aşağıdaki örnekte form yardımıyla kullanıcılarından adı, soyadı, hobileri, cinsiyeti, yaşı ve e-postası alınmıştır.

Tablo 3.24. Form Örneği

```
<!DOCTYPE HTML>
<html>
<head>
    <meta charset="UTF-8" />
    <title>Veri Girişi</title>
</head>
<body>
<form action="kontrol.php" method="get" >
    <table border="0">
        <tr>
            <td>Ad :</td>
            <td><input type="text" name="ad" id="ad" required
placeholder="Lütfen Adınızı Girin"></td>
        </tr> <tr>
            <td>Soyad :</td>
            <td><input type="text" name="soyad" id="soyad" required
placeholder="Lütfen Soyadınızı Girin"></td>
        </tr> <tr>
            <td colspan="2">
                FUTBOL<input type="checkbox" name="hobi1" id="hobi1">
                BASKETBOL<input type="checkbox" name="hobi2" id="hobi2">
                KARATE<input type="checkbox" name="hobi3" id="hobi3">
                MÜZİK<input type="checkbox" name="hobi4" id="hobi4">
            </td>
        </tr> <tr>
            <td colspan="2">
                KADIN<input type="radio" name="cinsiyet" id="cinsiyet" value="kadın">
                ERKEK<input type="radio" name="cinsiyet" id="cinsiyet" value="erkek">
            </td>
        </tr> <tr>
            <td>YAŞ:</td>
            <td><input type="number" name="yas" id="y" min="8" max="88"></td>
        </tr>
        <tr>
            <td>E-POSTA:</td>
            <td><input type="email" name="mail" id="mail" required></td>
        </tr>
        <tr>
            <td colspan="2">
                <input type="submit" name="kaydet" id="kaydet" value="KAYDET">
                <input type="reset" name="sil" id="sil" value="TEMİZLE">
            </td>
        </tr> </table>
    </form>
</body>
</html>
```



HTM5 ile gelen yeni etiketlerle etkileşimli formlar yapmak çok daha kolaydır.

```
</form>
</body>
</html>
```

Web Sayfası Görüntüsü

Ad :

Soyad :

FUTBOL BASKETBOL KARATE MÜZİK

KADIN ERKEK

YAŞ:

E-POSTA:

Örnek

- Bilgisayarımızı açarken veya e-posta adresimizi girerken form örnekleriyle karşılaşırız.

Tablo 3.25. Örnek Form

İş BAŞVURU FORMU	
Adı	Text kontrolü
Soyadı	Text Kontrolü
E-posta	Text Kontrolü
Sıfır	Password Kontrolü
Doğum Tarihi	Date kontrolü
Doğum Yeri	Select kontrolü
Cinsiyet	Radio Kutusu Bay, Bayan
Referanslar	Textarea nesnesi
Alışkanlıklar	<ul style="list-style-type: none"> - Sigara içiyorum, radio nesnesi - Sosyal içiciyim, radio nesnesi - İçmiyorum, radio nesnesi
Yabancı Dil	İngilizce, checkbox nesnesi Almanca, checkbox Fransızca, checkbox İspanyolca, checkbox Çince, checkbox Rusça, checkbox
Askerlik	Tecilli, Muaf, Yapıldı, Select nesnesi
Eğitim	Lise, Ön Lisans, Lisans, Yüksek Lisans, Doktora Select nesnesi
Geçmiş İşler	Textarea Kaydet butonu, Submit

Yukarıda başka bir form örneği görülmektedir. Bu form örneğinde de farklı veri tipi giriş örnekleri verilmiştir.



Bireysel Etkinlik

- Şimdi siz de Şekil 3.1.'deki gibi farklı veri girişi tiplerini içeren bir iş başvurusu form örneği hazırlayın.



•**GİRİŞ**

•Bu bölümde HTML5'in girdi (`input`) elemanlarından bahsedilecektir. Önceki haftalardan hatırlayacağınız üzere `<input>` etiketi, `<form>` ve `</form>` etiketleri arasında yer alır ve kullanıcılarından farklı tür veri girişi yapacağı alanları tanımlamaktadır. HTML5 ile beraber çok sayıda yeni veri giriş tipi ve form elemanı dâhil olmuştur. Yeni veri giriş tiplerinden sonra bu tiplere kazandırılan yeni özellikler anlatılacaktır.

•**HTML5 İLE YENİ VERİ GİRİŞ (INPUT) TIPLERİ**

•HTML5 ile veri girişi tipleri çeşitlilik göstermektedir. Aşağıda yeni veri giriş tiplerinden öne çıkanlar sıralanmıştır:

- color, renkler için kullanılır.
- email, e-posta için kullanılır.
- range, fiyat aralığı vb. yazmak için kullanılır (min, max, step ve value parametreleri alır).
- search, arama girdileri için kullanılır.
- number,sadece sayısal veri girişleri için kullanılır.
- tel, telefon yazmak için kullanılır.
- time, saat için kullanılır.
- url, url girebilmek için kullanılır.
- date, doğum günü, yıl dönümü vb. için kullanılır.
- datetime, date tipine ek olarak saat de kapsayacak şekilde kullanılır.
- month, ay bilgisini yazmak için kullanılır.
- week, hafta bilgisini yazmak için kullanılır.

•**HTML5 İLE YENİ FORM ÖZELLİKLERİ**

•HTML5 ile form elemanlarına yeni özellikler kazandırılmıştır. Özellikle `input` etiketiyle beraber kullanılan bu özellikler, formların kullanılaklığını artırmıştır. Aşağıda yeni eklenen özellikler verilmektedir:

- autocomplete, otomatik tamamlama özelliği için kullanılır.
- validate/novalidate, formun doğrulama yapılması/yapılmaması gereken durumlarda kullanılır.
- autofocus, sayfa yüklenliğinde otomatik olarak focus/odaklanma özelliği alır.
- form, veri girişini, sayfanın başka bir yerinde ki form'a dâhil etmek için kullanılır.
- formaction, form'a birden fazla action özelliği kazandırmak için kullanılır.
- formmethod, form'a birden fazla method özelliği kazandırmak için kullanılır.
- formnovalidate, form elemanları kontrol edilmeden gönderimi için kullanılır.
- formtarget, form verileri gönderildikten sonra nerede görüntüleneceğini gösteren bir isim ya da anahtar sözcük belirlemek için kullanılır.
- min ve max özellikleri number, range, date, datetime, datetime-local, month, time ve week gibi girdilerin en küçük ve en büyük değerlerini belirlemek için kullanılır.
- multiple, e-posta ve dosya seçimi gibi form elemanlarına birden fazla değer girilmesini sağlamak için kullanılır.
- pattern, text, search, url, tel, email, ve password gibi veri girişlerinin kontrollü yapılması sağlanabilmek için kullanılır.
- placeholder, bir giriş alanına beklenen değerini açıklayan kısa bilgi verilebilmek için kullanılır.
- required özelliği, form gönderilmeden önce istenen girdilerin doldurulmasını zorunlu kılmak için kullanılır.
- step, geçerli sayı aralığında veri girişi yapılmasını sağlamak için kullanılır.
- readonly, sadece okuma izin vermek için kullanılır.

•**SONUÇ**

•Bu bölümde HTML5'in yeni giriş (`input`) elemanları anlatılmıştır. HTML5 ile beraber çok sayıda yeni veri giriş tipi ve form elemanları dâhil olmuştur. Böylelikle etkileşimli formlar hazırlanabilmektedir.

:Özet

DEĞERLENDİRME SORULARI

1. Hangi iki tarayıcı HTML5 ile gelen yeni veri tiplerinin çoğunu desteler?
 - a) Internet Expolorer (IE) ve Firefox
 - b) IE ve Chrome
 - c) Safari ve Firefox
 - d) Opera ve Chrome
 - e) IE ve Safari

2. Aşağıdaki input veri girişi kod satırında ile yazılan yere hangi veri tipi gelmelidir?
`<input type="....." min="1" max="5">`
 - a) color
 - b) email
 - c) number
 - d) search
 - e) url

3. Aşağıdaki input veri girişindeki Atatürk yazısının sadece okumaya izin verilir, değişiklikle veya yeni veri girişine izin verilmeyecek şekilde olması için ile yazılan yere hangi veri tipi gelmelidir?
`<input type="text" value="Atatürk",>`
 - a) step
 - b) required
 - c) autocomplete
 - d) autofocus
 - e) readonly

4. Aşağıdaki form özelliklerinden hangisi, veri girişi için kullanıcıya ipucu ya da bazı detaylar hakkında bilgi vermek amaçlı kullanılır?
 - a) placeholder
 - b) required
 - c) validate
 - d) url
 - e) pattern

5. HTML5'te time etiketi ne yapar?
 - a) Saat söyler.
 - b) Doğum günü söyler.
 - c) Zamanımızı ayarlamaya yarar.
 - d) Günümüzü düzenli bölmeye yarar.
 - e) Veri girişi tipi değildir.

6. Aşağıdaki gibi bir veri girişi özelliği varsa hangi değer girilemez?

<input type="number" name="veri" step="3" min="-30" max="30">

- a) -3
- b) 9
- c) 12
- d) 16
- e) 27

7. Aşağıdakiler hangisi HTML5 ile gelen yeni veri tipleridir?

- a) Text, textarea, multiplebox, list
- b) Email, month, number, color, search
- c) Month, date, table, email, textarea
- d) Body, head, title, table, foot
- e) Autocomplete, input, form, action

8. Aşağıdakilerden hangisi Formtarget özelliğinden bahsetmektedir?

- a) form verisini farklı bir adrese farklı metot ile gönderme imkânı sunar.
- b) form verileri gönderildikten sonra nerede görüntüleneceğini gösteren bir isim ya da anahtar sözcük belirler.
- c) form gönderildiği zaman, girdi kontrolü ile farklı bir dosyaya gönderim imkânı sağlanabilir.
- d) form elemanları farklı formlarda da tanımlanabilir.
- e) form elemanları kontrol edilmeden gönderilir.

9. Form elemanlarının form dışında da tanımlanmasına izin vermek istenilirse hangi özelliğini kullanmak gereklidir?

- a) formaction
- b) formmethod
- c) formnovalidate
- d) formvalidate
- e) form

10. Aşağıdaki form elemanlarını içerecek kod satırında ile boş bırakılan yere hangi düğme(buton) tipi yazılmalıdır ki tıklandığında verileri sunucuya göndersin?

<input type="....." name="kaydet" value="KAYDET">

- a) reset
- b) action
- c) submit
- d) get
- e) post

Cevap Anahtarı

1.d, 2.c, 3.e, 4.a, 5.a, 6.d, 7.b, 8.b, 9.e, 10.c

YARARLANILAN KAYNAKLAR

HTML5 Tutorial. 7 Ağustos 2019 tarihinde <http://www.tutorialspoint.com//html5/index.htm> adresinden erişildi.

HTML5 Tutorial. 7 Ağustos 2019 tarihinde <https://www.w3resource.com/html5/introduction.php> adresinden erişildi.

HTML. 7 Ağustos 2019 tarihinde
<https://gelecegiyazanlar.turkcell.com.tr/konu/web-programlama/egitim/101> adresinden erişildi.

HTML 7 Ağustos 2019 tarihinde <https://html.spec.whatwg.org/multipage/> adresinden erişildi.

HTML5 İLE İLERİ TEKNOLOJİLER



İÇİNDEKİLER

- HTML5 ile Çoklu Ortam
- HTML5 ile Grafik
- HTML5 ile Konum Belirleme
- HTML5 ile Sürükle Bırak
- HTML5 ile Uygulama Önbellekleme
- HTML5 ile Gelen Diğer API'ler
- HTML5 İle Kaldırılan Etiketler



HEDEFLER

- Bu üniteyi çalıştıktan sonra;
 - HTML5 ile çoklu ortam ve grafik uygulamaları hazırlayabilecek,
 - HTML5 ile konum belirleyebilecek,
 - HTML5 ile sürükle bırak özelliğini kullanabilecek,
 - HTML5 ile uygulama önbellekleme ve diğer API'leri sayfalara entegre edebilecek,
 - HTML5 ile kaldırılan etiketleri sıralayabileceksiniz.

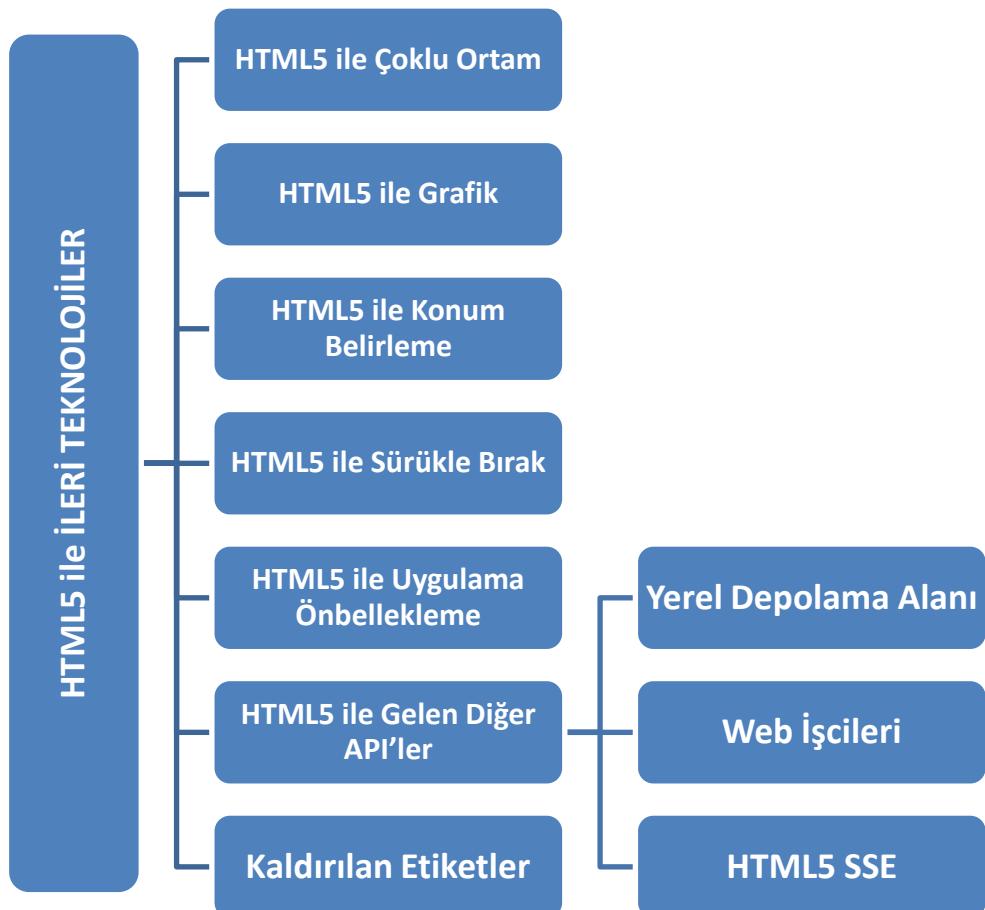


Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET PROGRAMCILIĞI II

Doç. Dr. İbrahim
ÇETİN

ÜNİTE
4



GİRİŞ

Piyasaya çıktıgı ilk günden beri web sayfalarının görsel tasarımda kullanılan HTML dili, 1980'li yıllarda günümüzde gelişerek, internet kullanıcılarına ve arama motorlarına daha faydalı hâle gelmeye çalışmaktadır. Son sürüm olarak HTML5'in piyasaya sürülmesiyle, alanda önemli değişiklikler ve yenilikler meydana gelmiştir ve bu bölüme kadar bu yeniliklerin bazlarından bahsedilmiştir. Bu bölümde ise HTML5'in en büyük yeniliklerinden biri olarak gösterilen, ileri teknoloji olarak söyleyebileceğimiz, web sayfalarındaki birçok yeni özelliğe kolaylıkla adapte olabilmesinden bahsedilecektir. W3C desteği ile hazırlanan bu yenilikler üzerinde hâlen çalışmaya devam edilmektedir. Yayınlanan raporlara göre bu yenilikler devam edecek ve çok daha fazla kullanıcıya ulaşacaktır.

Çoklu ortam desteği, grafik çizimler, konum belirleme özelliği, sürükle bırak uygulamaları, önbellekleme gibi güncel teknolojiler artık HTML5 ile hem kolay hem de hızlıca yapılmaktadır. HTML5'in yeni elementleriyle birkaç satırda video ve ses dosyaları sayfalardan paylaşılabilir, Javascript ve CSS desteği ile istenilen çizimlerle oyuncular yapılabilir, haritalar üzerinde konumlar belirlenebilir, sayfalar çevrimdışı durumunda da görüntülenebilir.

Bütün bu yeni özelliklerle HTML dili artık daha güçlü, daha etkili hâle gelmiştir. Statik sayfaların yapımında kullanılan ilk sürümlerinden, tasarımcıların ve programcıların istediği birçok şeyi kolaylıklaabilen, başka bir ifadeyle çok daha profesyonel sayfaların tasarımına olanak sağlayan bir dil hâline gelmiştir. Bu bölümde farklı ileri teknolojilerin HTML5 ile nasıl çalıştığı hakkında bilgiler verilecek, örnekler yapılarak anlatılacaktır.

HTML5 İLE ÇOKLU ORTAM



HTML5 ile çoklu ortam dosyaları farklı eklentilere ihtiyaç duymadan web sayfalarına eklenebilmektedir.

HTML5 ile çoklu ortam dosyaları ses, video gibi, Silverlight ya da Flash gibi eklentilere ihtiyaç duymadan web sayfalarına eklenebilmektedir. HTML5 ile gelen bu yeni standardın yaygın olarak bilinen çoklu ortam etiketleri şunlardır: video, audio, embed ve object.

<video> etiketi ile bir video web sayfasında yayınlanabilmektedir. Bu etiketin farklı parametreleri vardır.

- src: İlgili dosyanın kaynağını belirtmek için kullanılır.
- controls: "play, pause, ses" gibi denetimler eklenir.
- width ve height: Videonun genişlik ve yükseklik ayarlarını belirler.

Aşağıdaki örnekte 320*240 olarak girilmiştir.

Tablo 4.1. Video Etiketi

HTML5 Kodları

```
<video width="320" height="240" controls>
<source src="movie.mp4" type="video/mp4">
<source src="movie.ogg" type="video/ogg">
```

Tarayıcınız bu özelliği desteklemiyor.

</video>

Web Sayfası Görüntüsü

Yukarıdaki örneğin çalışabilmesi için hazırladığınız web sayfasıyla video dosyasının aynı dizinde olması gerekmektedir. Eğer internetten farklı bir video dosyası oynatmak istenirse src ile dosya yolu şu şekilde değiştirilebilir.



:Ornek

```
•<source src="https://www.w3schools.com/html/mov_bbb.mp4"
type="video/mp4" />
```

Tarayıcı öncelikle mp4 dosyasını göstermeye çalışır, daha sonra ogg dosyasını, eğer ikisini de gösteremiyor ise bu özelliği desteklemediği yazılır. Başka bir ifadeyle, video etiketini desteklemeyen tarayıcılar için uyarı eklemek gerekmektedir. HTML5 şu anda MP4, OGG ve WebM video dosya biçimlerini farklı tarayıcılar tarafından desteklemektedir. Şimdilik desteklenen 3 farklı formatın codec bilgileri ve tarayıcı destekleri aşağıda verilmiştir:

- MP4: H264 video codec ve AAC audio codec
- WebM: VP8 video codec ve Vorbis audio codec
- Ogg: Theora video codec ve Vorbis audio codec

Tablo 4.2. Tarayıcıların Desteklediği Video Formatları

Tarayıcı	MP4	WebM	OGG
Firefox	✓	✓	✓
Chrome	✓	✓	✓
Safari	✓		
Opera		✓	✓
Internet Explorer	✓		



HTML5 şu anda MP4, OGG ve WebM video dosya biçimlerini desteklemektedir.



- Youtube üzerinden istediğiniz bir videoyu kendi hazırladığınız sayfa içinde oynatacak şekilde hazırlayınız. Video etiketinde controls parametresinin olmasına dikkat ediniz.

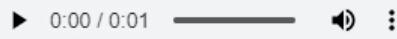


HTML5 şu anda MP3, WAV ve OGG ses dosya biçimlerini desteklemektedir.

HTML5 Kodları

```
<audio controls>
<source src="horoz.ogg" type="audio/ogg">
<source src="horoz.mp3" type="audio/mpeg">
Tarayıcınız audio elementini desteklemiyor.
</audio>
```

Web Sayfası Görüntüsü



Tarayıcıların destekledikleri ses formatları Tablo 4.4.'te verilmiştir. Görüldüğü üzere mp3 formatı bütün tarayıcılar tarafından desteklenmektedir.

Tablo 4.4. Tarayıcıların Desteklediği Ses Formatları

Tarayıcı	MP3	WAV	OGG
Firefox	✓	✓	✓
Chrome	✓	✓	✓
Safari	✓	✓	
Opera	✓	✓	✓
Internet Explorer	✓		

<embed> etiketi ile de web sayfalarına ses, video ve animasyon eklenebilmektedir. Aşağıdaki örnekte video/müzik oynatıcı programın arayüzü görünecektir. Bu sayede kullanıcı oynatma, durdurma, ses gibi ayarları yapabilir. Benzer şekilde *<object>* ile de ses ve video gibi dosyalar eklenebilmektedir.

Tablo 4.5. Embed ve Object Etiketleri

HTML5 Kodları

```
<embed src="muzikDosyasi.mp3" autostart="false" />
<embed src="videoDosyasi.avi" autostart="false" />
<object width="420" height="315"
      data="https://www.youtube.com/embed/tgbNymZ7vqY">
</object>
```

HTML5 İLE GRAFİK



HTML5 ile grafik işlemleri canvas ve svg etiketleriyle yapılmaktadır.

HTML5 ile grafik yönün kuvvetlenmesi için **<canvas>** ve **<svg>** etiketleri geliştirilmiştir. **<canvas>** etiketi ile sayfada bir tuval alanı (dikdörtgen alanı) oluşturur. Tuvale çizim JavaScript dili ile yapılmaktadır. Bu etiket yardımıyla grafikler çizilebilir, fotoğraf kompozisyonları yapılabilir veya animasyonlar hazırlanabilmektedir. Aşağıdaki örnekte görüldüğü üzere canvas ile eni 200 yüksekliği 100 piksel olan bir dikdörtgen çizilmiştir.

Tablo 4.6. Canvas Etiketi

HTML5 Kodları

```
<canvas id="canvasim" width="200" height="100"
style="border:1px solid red;">
</canvas>
```

Web Sayfası Görüntüsü



Örnek

- <canvas id="Tuvalim" width="200" height="100"></canvas>

Aşağıdaki örnekte ise çizilen canvas'ın içi doldurulup dikdörtgen hâline getirilmiştir. Bu örnek için küçük JavaScript kodları kullanılmıştır. JavaScript dersleri ilerleyen haftalarda daha detaylı anlatılacaktır.

Tablo 4.7. Canvas Etiketinin JavaScript ile Kullanımı

HTML5 Kodları

```
<canvas id="ilkCanvas"> </canvas>
<script>
  var c1 = document.getElementById("ilkCanvas");
  var c2 = c1.getContext("2d");
  c2.fillStyle = 'blue';
  c2.fillRect(0,0,150,75);
</script>
```

Web Sayfası Görüntüsü



Aşağıdaki başka bir canvas örneğinde ise bayrak çizilmiştir.

Tablo 4.8. Canvas Etiketiyle Bayrak Çizimi

HTML5 Kodları

```
<canvas id="myCanvas" width="360" height="210" style="border:1px solid #c3c3c3;"> </canvas>
<script type="text/javascript">
    var c=document.getElementById("myCanvas");
    var ctxt=c.getContext("2d");
    ctxt.fillStyle="#FF0000";
    ctxt.beginPath();
    ctxt.arc(150,105,60,0,Math.PI*2,true);
    ctxt.closePath();
    ctxt.fill();
    var ctxt2=c.getContext("2d");
    ctxt2.fillStyle="#FFF";
    ctxt2.beginPath();
    ctxt2.arc(168,105,48,0,Math.PI*2,true);
    ctxt2.closePath();
    ctxt2.fill();
</script>
```

Web Sayfası Görüntüsü



SVG, Ölçeklenebilir
Vektör Grafikleri
anlamına gelmektedir.



HTML5 ile gelen bir diğer grafik etiketi ise SVG'dir. *SVG'nin açılımı Scalable Vector Graphics (Ölçeklenebilir Vektör Grafikleri)* olarak ifade edilir. Başlıca özellikleri JavaScript kullanmadan vektör temelli grafikler çizilebilmesi, grafiklerin XML formatında tanımlanması ve grafiklerin yeniden boyutlandırılsa veya büyütülüp küçültülse bile kalitelerinin düşmemesidir. Aşağıdaki örnekte SVG kod etiketiyle farklı geometrik şekiller çizilmektedir.

Tablo 4.9. SVG Etiketi

HTML5 Kodları

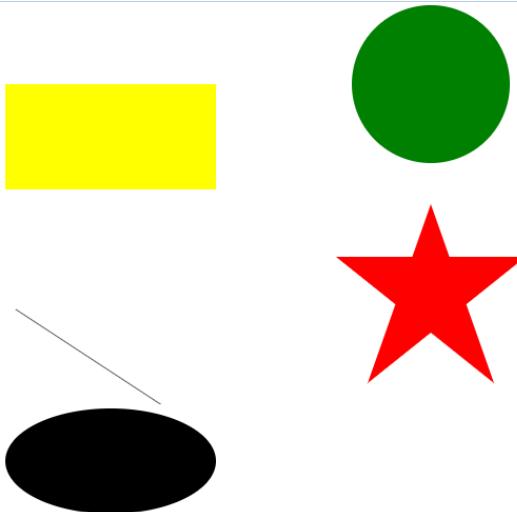
```
<!-- dikdörtgen çiziyoruz -->
<svg height="100">
<rect width="200" height="100" fill="yellow" />
</svg>
```



SVG ile farklı çizimler yapılabilir.

```
<!-- daire çiziyoruz -->
<svg height="200">
<circle cx="100" cy="100" r="75" fill="green" />
</svg>
<!-- çizgi çiziyoruz --><br>
<svg height="100">
<line x1="10" y1="10" x2="300" y2="200" style="stroke: black; stroke-width:20"/>
</svg>
<!-- polygon çiziyoruz -->
<svg height="200">
<polygon points="100,10 40,180 190,60 10,60 160,180" fill="red"/>
</svg>
<!-- elips çiziyoruz --><br>
<svg height="200">
<ellipse cx="100" cy="50" rx="100" ry="50" />
```

Web Sayfası Görüntüsü



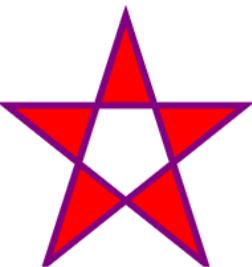
Aşağıdaki örnek SVG kod satırlarıyla polygon yıldızı farklı renklerle çizilmektedir.

Tablo 4.10. SVG Etiketiyle Polygon Yıldızı

HTML5 Kodları

```
<svg width="300" height="200">
<polygon points="100,10 40,198 190,78 10,78 160,198"
style="fill:red;stroke:purple;stroke-width:5;fill-rule:evenodd;" />
</svg>
```

Web Sayfası Görüntüsü



Canvas ve SVG etiketleri bazen birbirile karıştırılmaktadır. Fakat özellikleri açısından farklılıklar gösterirler. Aşağıdaki tabloda Canvas ve SVG arasındaki farklar karşılaştırılmıştır.

Tablo 4.11. Canvas ve SVG Etiketleri

Canvas	SVG
<ul style="list-style-type: none"> Çizimler çözünürlüğe bağlı. JavaScript olayları desteklemez. .png veya .jpg olarak kaydedebilir. Grafiksel oyunlar için uygun. 	<ul style="list-style-type: none"> Çizimler çözünürlükten bağımsız. JavaScript olayları destekler. .png veya .jpg olarak kaydedilemez. Yoğun grafiksel oyunlar için uygun değil.



Bireysel Etkinlik

- Yukarıdaki Tablo 4.8.'deki bayrak örneğini renklerini değiştirerek tekrar hazırlayın. Örneğin bayrak kırmızı, içindeki hilal beyaz olsun. Bir sonraki aşamada ise hilalin yanına yıldız koymaya çalışın.

HTML5 İLE KONUM BELİRLEME

Geolocation API ile konum bilgilerini almak ve alınan bilgileri harita üzerinde göstermek mümkündür. HTML5'te, Geolocation API kullanarak, sayfaları ziyaret edenlerin konumları alınıp web sayfası üzerinden paylaşılabilir. Kullanıcıların gizlilik hakları ihlal edilebileceğinden öncelikle onay alınarak bilgiler işlenir.

Aşağıdaki örnekte JavaScript kullanılarak navigator.geolocation ve getCurrentPosition metodları ile konum bilgileri alınmaktadır. Benzer JavaScript konuları ilerleyen haftalarda daha detaylı anlatılacaktır.



Konum belirleme özelliğini GPS'i olan cihazlarda kullanmak gereklidir.

Tablo 4.12. HTML5 ile Konum Belirleme

HTML5 Kodları
<pre><script> function konumuGetir() { if (navigator.geolocation) { navigator.geolocation.getCurrentPosition(konumBilgileri); } else { document.write("Tarayıcınız Geolocation API desteklemiyor."); } function konumBilgileri(k) { document.write("Enlem: " + k.coords.latitude); document.write("
"); document.write("Boylam: " + k.coords.longitude); } konumuGetir(); </script></pre>
Web Sayfası Görüntüsü
<p>Enlem: xxxxxx Boylam: yyyy // bu örnek çıktıda x ve y değerleri bulunduğuuz enlem ve boylam değerleri olarak alınacaktır.</p>

Yukarıdaki örneğin kısaca açıklaması:

- Öncelikle tarayıcının, Geolocation API destekleyip desteklemediği kontrol ediliyor. Eğer desteklemiyorsa “Tarayıcınız Geolocation API desteklemiyor.” mesaj yazdırılıyor.
- getCurrentPosition() metoduna konumBilgileri fonksiyonunun değerleri gönderildi.
- konumBilgileri fonksiyonu içerisinde enlem ve boylam bilgileri ekrana yazıldı.



Örnek

• Konum belirleme örneğini harita (Google Map) üzerinde görmek isterseniz,
https://www.w3schools.com/html/html5_geolocation.asp adresinden ulaşabilirsiniz.

HTML5 İLE SÜRÜKLE BIRAK

Sürükle bırak (Drag & Drop) yöntemi görsel ortamlarda sıkça kullanılan özelliklerdir. Bir nesneyi, metni veya resmi başka bir alana fare yardımıyla taşımayı sağlamaktadır. HTML5 ile bu özellik daha kolay hâle gelmiştir ve tüm

nesnelere uygulanabilmektedir. Aşağıdaki örnekte bir logo sürükleme etrafı çizgili olan alana bırakılabilmektedir.

Tablo 4.13. HTML5 ile Sürükle Bırak

HTML5 Kodları

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8" />
<title>HTML5 Sürükle Bırak</title>
<style>
#alan {
    width: 250px;
    height: 250px;
    padding: 10px;
    border: 1px solid #cccccc;
}
#resim {
    border: 1px solid red;
}
</style>
<script>
function surukle(o) {
    o.dataTransfer.setData("text", o.target.id);
}
function birak(o) {
    o.preventDefault();
    var veri = o.dataTransfer.getData("text");
    o.target.appendChild(document.getElementById(veri));
}
</script>
</head>
<body>
<div id="alan" ondrop="birak(event)" ondragover="return false;"></div>
<br />

</body>
</html>
```



HTML5 ile ileri teknolojiler JavaScript ve CSS desteğiyle kullanılmaktadır.



Draggable, sürüklemeye açmak veya kapatmak için kullanılır.

Bu örnekte hem CSS hem de JavaScript kodları kullanılmıştır. Ondragover ve ondragstart gibi sürükle bırak işlemlerinin yapılmasında kullanılan fonksiyonlar çağırılmıştır. Bu ve benzeri nesnelerin kullanımıyla ilgili detaylar ilerleyen haftalarda anlatılacaktır. Yukarıdaki örneğin çalışma sürecini kısaca özetlemek gerekirse, img yani resim (bu örnekte logo) fare ile sürüklenebilir durumda (özellik이 true olarak verilmiş), resim sürüklenemeye başladığında surukle()

fonksiyonu çalıştırılıyor, bırakıldığından ise örnekte “alan” olarak tanımlanmış div bölgесine bırak() fonksiyonu çağırılıyor. Böylelikle sürükle bırak uygulaması çalıştırılabilir. Sürükle bırak özelliğini kullanarak farklı uygulamalar, özellikle oyunlar, etkileşimli etkinlikler yapılmaktadır. Bu özelliği kullanabilmek için JavaScript'in farklı fonksiyonlarından yararlanılmaktadır. Bazı fonksiyonlar ve görevleri şöyle sıralanabilir:

- draggable: Doğru/Yanlış (True/False) değeri alır, sürüklemeye açmak veya kapatmak için kullanılır.
- ondragstart: Sürüklemenin başladığı anı yakalar ve sürüklenen nesneye verilir.
- ondragover: Sürüklenen bir nesne ile üzerine geldiği anı yakalar.
- ondragleave: Sürüklenen bir nesne ile üzerinden çıktıığı anı yakalar.
- ondrop: Sürüklenen nesnenin üstüne bırakıldığı anı yakalar.
- ondragenter: Sürüklenen bir nesne ile üzerinde hareket edildiği anı yakalar.
- ondragend: Sürüklemenin bittiği anı yakalar.



Bireysel Etkinlik

- İstedığınız bir resmi (örneğin width="150" height="150") istediğiniz bir div alanına sürükleip bırakacak şekilde sayfa hazırlayın. Yukarıdaki örnekte yer alan JavaScript fonksiyonlarını kullanabilirsiniz.


HTML5'te önbellekleme ile çevrimdışı durumunda bile web sayfalarında dolaşım sağlanabilmektedir.

HTML5 İLE UYGULAMA ÖNBELLEKLEME

Bir web sayfasının/uygulamasının bir kez **önbelleğe** (cache) atılınca, çevrimdışı durumunda da çalışabilir olmasına HTML5'te "Uygulama Önbellekleme (Application Cache)" denir. Web uygulamasının bir çevrimdışı sürümünü yapmak için bir **manifest** dosyası oluşturmak ve manifest sınıfını (attribute) belgenin HTML etiketine eklemek yeterlidir. Dosyaların önbelleklenmesi kullanıcılarla iyi bir hız deneyimi yaşatır. Sunucu tarafındaki yükü ise azaltacaktır çünkü sayfalara sadece güncellenen/eklenen veriler çekilecektir. Aşağıdaki örnekte html etiketi içine manifest parametresi eklenerek Uygulama Önbellekleme kullanabilir duruma getirilmiştir.

Tablo 4.14. HTML5 ile Uygulama Önbellekleme

HTML5 Kodları

```
<!DOCTYPE html>
<html manifest="uygulama.appcache">
  <body>
    Döküman içeriği...
  </body>
```

</html>

Manifest dosyası, basit bir metin dosyasıdır. Uygulamanın hangi parçasının önbelleğe alınıp alınmayacağı belirler. Örnek bir manifest dosya içeriği aşağıda verilmiştir.

Tablo 4.15. Manifest Dosyası

HTML5 Kodları

CACHE MANIFEST
index.php
stylesheet.css
images/logo.jpg
scripts/help.js

HTML5 İLE GELEN DİĞER API'LER



HTML5 ile çok sayıda API desteği gelmiştir.

HTML5 ile beraber çok sayıda API desteği gelmiştir. Bunlardan birkaçı bu hafta inceleneciktir.

Yerel Depolama Alanı (Local Web Storage)

HTML5 ile beraber yerel depolama birimleri ile kullanıcı tarayıcısına yerel olarak veri depolanabilmektedir. Bu bilgiler daha önce cerezler (cookies) ile saklanmaktaydı. Fakat cerez kullanımında, kullanıcıların bilgisayarına sızma veya kullanıcı bilgilerine ulaşma yapılabildiğinden, bu sağlıklı bir yöntem değildi. Local Storage kavramı ile bilgiler kullanıcının tarayıcısında saklanır ve cerezlere göre daha güvenli ve hızlıdır. İki tip web depolama versiyonu vardır: Local Storage, Session Storage.

Local Storage, JavaScript fonksiyonu olan window.localStorage ile, kaydedilen bilgileri, belirtilmeyen herhangi bir tarihe kadar saklanabilmektedir. Bu bilgiler istenildiği zaman silinebilmektedir.

Tablo 4.16. HTML5 ile Yerel Depolama Alanı

HTML5 Kodları

```
<script type="text/javascript">  
// verimizi, v1 adıyla tarayıcımıza saklayalım.  
localStorage.setItem("v1", "Atatürk Üniversitesi");  
// ekrana yazdırıralım.  
document.write(localStorage.getItem("v1"));  
</script>
```



HTML5'in yerel depolama API'si ile daha güvenli sayfalar hazırlanabilmektedir.

Session Storage ise yine JavaScript fonksiyonu olan window.sessionStorage ile, istenilen bilgiler sadece bir oturum (session) için saklanmaktadır. Başka bir ifadeyle web sayfası kapatıldığı zaman ya da tarayıcıyı kapatıldığı zaman bu bilgiler tarayıcı belleğinden silinmektedir.

Tablo 4.17. HTML5 ile SessionStorage**HTML5 Kodları**

```
<script type="text/javascript">
// verimizi, v1 adıyla tarayıcımıza saklayalım.
sessionStorage.setItem ("v1", "Ogrenci1");
// ekrana yazdırıralım.
document.write(sessionStorage.getItem("v1"));
</script>
```

Web İşçileri (Workers)

Web sayfalarında JavaScriptler, tek işte çalıştırılacak şekilde tasarlanmıştır, yani birden çok komut aynı anda çalıştırılamaz. JavaScript, CPU kullanımının yüksek olduğu durumlarda tarayıcının kullanıcıya cevap veremediği durumlara gelebilmektedir. Web işçi (Web workers), sayfanın performansını etkilemeden diğer script'lerden bağımsız olarak arka planda çalışan bir JavaScript kodudur. Web işçileri kullanarak tıklama, nesneleri seçme gibi istenilen başka her türlü şeyi yapmaya devam edilebilmektedir. Bu API'yi kullanmak, çok çekirdekli işlemcilere sahip bilgisayarlarda daha etkilidir.

HTML5 SSE (Server-Sent Events- Server-Sent Olayları):

HTML5 Server-Sent Olayları, Web sayfasında kullanılmak üzere sunucudan güncellemeleri almak için kullanılır. Bu özellikle daha önce de mümkündü, ancak web sayfası herhangi bir güncelleme olup olmadığını sormak zorunda kalıyordu. Sosyal medya mesajları (Facebook/Instagram gibi), döviz fiyatları, haberler gibi güncellemeler örnek olarak verilebilir. Gelişmiş tarayıcıların hepsi bu özelliğe destekler.



HTML5 ile bazı etiketler kaldırılmış veya yerine yenileri eklenmiştir.

HTML5 İLE KALDIRILAN ETIKETLER

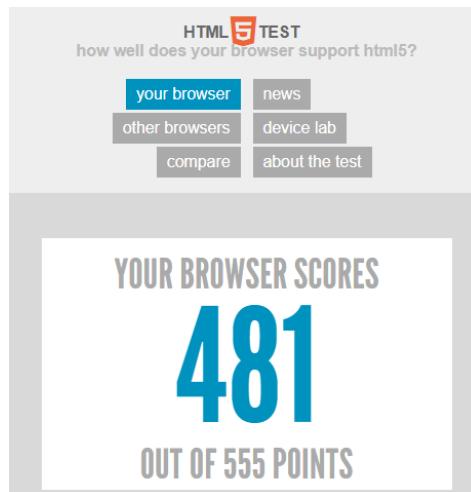
HTML5 ile beraber bazı etiketler artık kullanılmamaya başlanmıştır. Bu etiketlerin yerine alternatif, daha kullanışlı etiketler ya da yöntemler eklenmiştir. Aşağıdaki tabloda bu etiketlerden örnekler verilmiştir, yanlarına da yerine kullanılanlardan bahsedilmiştir.

Tablo 4.18. Kaldırılan Etiketler

Kaldırılan etiketler	Yerine kullanılanlar
<acronym>	<abbr>
<applet>	<object>
<basefont>	CSS içerisinde
<big>	
<center>	
<tt>	
	
<frame>	<iframe> ile
<frameset>	

<noframes>	
<strike>	
<dir>	

HTML5 etiketlerinin geliştirme aşamasının daha devam ettiğini biliyoruz. Hatta birçok internet tarayıcısı HTML5’i desteklemek için de çalışmalarına devam etmektedir. Kullandığınız web tarayıcınızın ne seviyede HTML5 desteklediği <http://html5test.com> adresine tarayıcınızdan erişerek öğrenilebilir. Bu web sitesi hangi tarayıcı ile siteye ulaşılırsa HTML5 ve ilgili özellikler için test uygulayarak tarayıcının bu özellikleri ne derece desteklendiğine dair puanlama yapmaktadır. Örnek bir çıktı aşağıdaki şekilde verilmektedir.



Şekil 4.1. HTML5 test sonucu

Sonuç olarak HTML5 ile ilgili gelişmeleri incelediğimizde, piyasaya sürüldüğünden itibaren çok ses getiren bir teknoloji olduğu görülmektedir. Özellikle bundan sonraki bölümlerde anlatılacak olan CSS ve JavaScript teknolojileriyle beraber kullanılması durumunda HTML5 hızla yaygınlaşmaya devam edecektir diyebiliriz. HTML5'in bizlere sağladığı temel kazanımları 3 maddede özetleyeceğiz;

- Daha fazla işlevsellik,
- Daha fazla semantik/anlamsallık,
- Daha fazla etkileşim.

Bunlara ilaveten güncel teknolojilerin adapte edilmesi, çapraz platform olacak şekilde tasarlanması ve mobil platformlara uyarlanabilir olması, gelecekte de kullanılacağıının işaretlerindendir. Bütün bunlara ilaveten HTML5’in bir diğer kullanım alanı ise oyunlardır, oyun sektörünün hızla büyüğü günümüzde internetteki çok sayıda sitede HTML5 ile geliştirilen oyunlar paylaşılmaktadır. Hem kodlarının paylaşıldığı hem de oyunların oynanıldığı birkaç siteyi şöyle sıralayabiliriz:

- W3Schools: https://www.w3schools.com/graphics/game_intro.asp
- HTML5 ile yapılmış 15 açık kaynaklı oyun: <https://superdevresources.com/open-source-html5-games/>
- En popüler HTML5 oyunları: <https://itch.io/games/html5>



W3School sitesi ile HTML5 öğrenebilir, hazırlanan oyunları inceleyebilirsiniz.



Özet

•**GİRİŞ**

- HTML5'in en büyük yeniliklerinden biri ileri teknoloji olarak söyleyebileceğimiz web sayfalarındaki birçok yeni özelliğe kolaylıkla adapte olabilmesidir. Çoklu ortam desteği, grafik çizimler, konum belirleme özelliği, sürükle bırak uygulamaları, önbellekleme gibi güncel teknolojiler artık HTML5 ile hem kolay hem de hızlıca yapılmaktadır. Bu bölümde farklı ileri teknolojilerin HTML5 ile nasıl çalıştığı hakkında bilgiler verilecek, örnekler yapılarak anlatılacaktır.

•**HTML5 İLE ÇOKLU ORTAM**

- HTML5 ile çoklu ortam dosyaları ses ve video gibi web sayfalarında yayılmak için kullanılan en yaygın etiketler aşağıda verilmiştir:
- <video> etiketi ile bir video web sayfasında yayınlanabilmektedir.
- <audio> etiketiyle de ses ve şarkılar web sayfalarından yayınlanabilir.
- <embed> ve <object> etiketleriyle de web sayfalarına ses, video ve animasyon eklenebilmektedir.

•**HTML5 İLE GRAFİK**

- HTML5 ile gelen grafik etiketleri şunlardır:
- <canvas> etiketi ile sayfada bir tuval alanı (dikdörtgen alanı) oluşturur. Bu etiket yardımıyla grafikler çizilebilir, fotoğraf kompozisyonları yapılabilir veya animasyonlar hazırlanabilir.
- <svg> etiketi ile vektör temelli grafikler çizilebilir, grafikler yeniden boyutlandırılırken veya büyütülüp küçültülürken kaliteleri düşmez.

•**HTML5 İLE KONUM BELİRLEME**

- Geolocation API ile konum bilgilerini almak ve alınan bilgileri harita üzerinde göstermek mümkündür.

•**HTML5 İLE SÜRÜKLE BIRAK**

- HTML5 ile sürüle bırak, yani metni veya resmi başka bir alana fare yardımıyla taşımak daha kolay hâle gelmiştir ve tüm nesnelere uygulanabilmektedir.

•**HTML5 ile Uygulama Önbellekleme**

- HTML5 ile web sayfalarının önbelleğe atılıp, çevrimdışı durumunda da çalışabilir hâle getirilmesi, yani önbellekleme yapılması sayfalar arasında dolaşımı hızlandırmıştır.

•**HTML5 ile Gelen Diğer API'ler**

- HTML5 ile beraber çok sayıda API desteği gelmiştir. En fazla dikkat çeken API'ler ise Yerel Depolama Alanı, Web (Workers) İşçileri, SSE (Server-Sent Events)'dir.

•**HTML5 İLE KALDIRILAN ETİKETLER**

- HTML5 ile beraber bazı etiketler artık kullanılmamaya başlanmıştır. Kaldırılan etiketler şunlardır: <acronym>, <applet>, <basefont>, <big>, <center>, <tt>, , <frame>, <frameset>, <noframes>, <strike>, <dir>

•**SONUÇ**

- Sonuç olarak HTML5 ile ilgili gelişmeleri incelediğimizde, piyasaya sürüldüğünden itibaren çok ses getiren bir teknoloji olduğu görülmektedir.

DEĞERLENDİRME SORULARI

1. I. Çizimlerle oyunlar yapılabilir.
II. Haritalar üzerinde konumlar belirlenebilir.
III. Sayfalar çevrimdışı durumunda çalışabilir.
Yukarıdakilerden hangisi ya da hangileri HTML5'in getirdiği yeniliklerdendir?
 - a) Sadece I
 - b) Sadece II
 - c) I ve II
 - d) I, II, III
 - e) II ve III
2. Aşağıdakilerden hangisi HTML5'in çoklu ortam ile gelen etiketlerinden değildir?
 - a) head
 - b) video
 - c) audio
 - d) embed
 - e) object
3. Aşağıdaki kod satırlarında video etiketiyle ilgili ... ile boş bırakılan yere ne gelmelidir?

```
<video>  
  <..... src="movie.mp4" type="video/mp4">  
</video>?
```

 - a) control
 - b) source
 - c) width
 - d) audio
 - e) mp4
4. Aşağıda web sayfası görüntüsü verilen etiket hangisidir?

 - a) head
 - b) video
 - c) audio
 - d) canvas
 - e) svg

5. Aşağıdaki HTML5 etiketlerinden hangisiyle vektörel grafik çizimler yapılabilir?
 - a) <canvas>
 - b) <svg>
 - c) <script>
 - d) <circle>
 - e) <polygon>
6. etiketi ile sayfada bir tuval alanı (dikdörtgen alanı) oluşturur. Boş bırakılan yere aşağıdaki hangi etiket gelmelidir?
 - a) <canvas>
 - b) <svg>
 - c) <script>
 - d) <circle>
 - e) <polygon>
7. <rect width="200" height="100" fill="yellow" /> kod satırı sizce çıktı olarak hangi geometrik şekli gösterir?
 - a) Elips
 - b) Polygon
 - c) Çizgi
 - d) Daire
 - e) Dikdörtgen
8. I. Sürükle bırak
II. Konum belirleme
III. Yerel Depolama Alanı
HTML5 ile gelen yeni API'lerden hangisi ya da hangileri JavaScript ile beraber kullanılmaktadır?
 - a) Yalnız I
 - b) Yalnız II
 - c) I ve II
 - d) I, II, III
 - e) II ve III
9., web sayfası kapatıldığı zaman ya da tarayıcı kapatıldığı zaman bu bilgiler tarayıcı belleğinden silinmektedir. Boş bırakılan yere hangi API gelmelidir?
 - a) Local Storage
 - b) Session Storage
 - c) Web Workers
 - d) SSE
 - e) JavaScript

10. Aşağıdakilerden hangisi kaldırılan `<frame>` ya da `<frameset>` etiketlerinin yerine kullanılan etiketlerdir?

- a) `<abbr>`
- b) `<object>`
- c) `<iframe>`
- d) `<object>`
- e) `<applet>`

Cevap Anahtarı

1.d, 2.a, 3.b, 4.c, 5.b, 6.a, 7.e, 8.d, 9.b, 10.c

YARARLANILAN KAYNAKLAR

HTML Dersleri. 7 Ağustos 2019 tarihinde <http://www.w3.org/TR/html5/> adresinden erişildi.

HTML5 & CSS. 6 Ağustos 2019 tarihinde
<https://gelecegiyazanlar.turkcell.com.tr/konu/web-programlama/egitim/201> adresinden erişildi.

HTML5 Dersleri. 6 Ağustos 2019 tarihinde <https://www.yusufsezer.com.tr/html5-dersleri/> adresinden erişildi.

HTML5. 6 Ağustos 2019 tarihinde <http://www.html-5-tutorial.com/> adresinden erişildi.

CSS VE METİN STİLLERİ



içindekiler

- CSS Nedir?
- CSS Sürümleri Nelerdir?
- CSS Tanımlama Yöntemleri
- CSS Seçicileri
- CSS ve Metin



HEDEFLER

- Bu üniteyi çalıştından sonra;
- CSS'in ne olduğunu tanımlayabilecek,
- CSS'in ne işe yaradığını maddeler hâlinde sıralayabilecek,
- CSS'in sürümlerini ve güncel sürümü kavrayabilecek,
- CSS tanımlama yöntemlerini öğrenebilecek,
- CSS seçicilerinin neler olduğunu ifade edebilecek,
- Metinlerle ilgili CSS tanımlarının sonuçlarını gösterebilecek,
- Metin biçimlerine bakarak uygun CSS tanımlarını seçebileceksiniz.



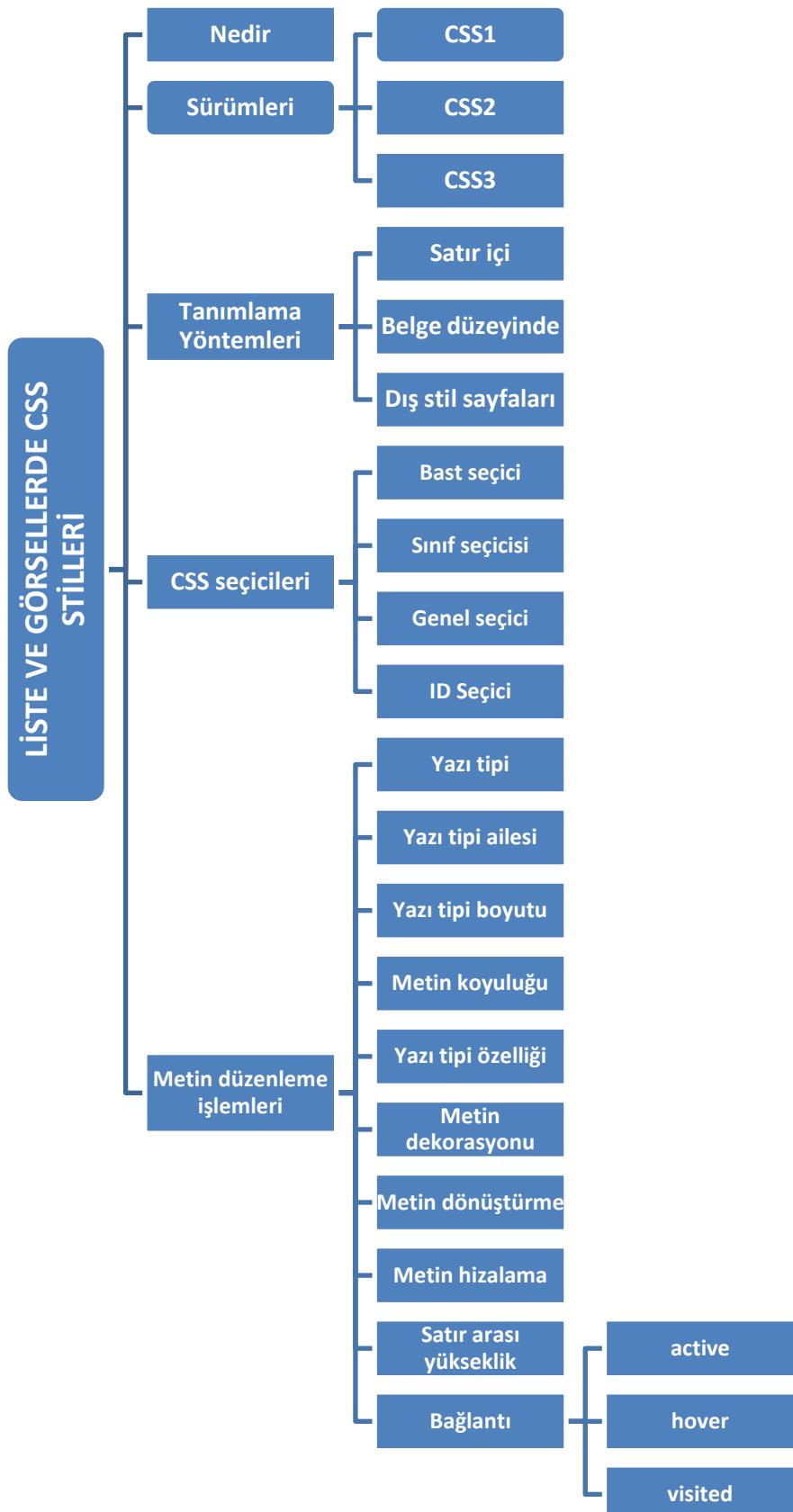
Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET PROGRAMCILIĞI II

Doç. Dr. Ercan TOP

ÜNİTE

5



GİRİŞ

Web sayfalarının oluşturulmasında kullanılan standart dil olan HTML5 ile ilgili konular önceki bölümlerde incelenmiştir. Bu bölümde öncelikle sayfaları şekillendirmek için kullanılan, etiketleri yazma zorluğunu ve karmaşasını azaltan CSS (Cascading Style Sheets- Basamaklı Stil Sayfaları) kavramından bahsedilecektir. Stil sayfalarının oluşturulduğu kural yapıları inceleneciktir. CSS'in sürümleri hakkında kısa bilgiler verilecektir. Stil tanımlamanın üç farklı yöntemi olan satır içi, sayfa düzeyinde ve dış sayfa stilleri örneklerle ve ekran görüntüleriyle gösterilecektir. Stil sayfalarının kurallardan olduğu ve her bir kuralın seçici ve bildirim blokundan olduğu vurgulanacaktır. Seçici türleri olan basit seçici, class (sınıf seçicisi), genel seçici ve ID seçici kavramları açıklanacak, uygulama örnekleriyle somutlaştırılmaya çalışılacaktır.

CSS tanıtımı, CSS kural yapıları, CSS tanımlama yöntemleri ve CSS seçicileri konuları incelendikten sonra bölümün geri kalanında CSS kullanarak metin düzenlemeyle ilgili sık kullanılan özellikler ve bu özelliklere atanabilecek değerlerden bahsedilecektir. Metinlerle ilgili incelenecek özellikler; yazı tipleri (font), yazı tipi aileleri (font-family), yazı tipi boyutu (font-size), metin hizalama (text-align), metin dekorasyon (text-decoration), metin ağırlığı (font-weight), yazı tipi özelliği (font-style), metin dönüştürme (text-transform), bağlantı kullanımı (link) ve satır yüksekliği (line-height) olarak belirlenmiştir. Her bir özellik hakkında önce açıklayıcı bilgiler verilecektir. Sonrasında ise özelliklerin kullanım şekilleri ve yöntemleri hakkında detaylı açıklamalar yapılacak, kullanım şekilleri ve yöntemleri ile ilgili tam HTML dosyaları veya HTML dosyalarının bazı bölümlerinin ekran görüntüleri paylaşılacaktır. Hazırlanan HTML kodlarının tarayıcılarında açıldığında ortaya çıkan görüntüleri de HTML dosyalarının içerikleriyle birlikte verilerek, öğrencilerin HTML kodu ve bu kodlarla ortaya çıkacak görüntüyü zihinlerinde eşleştirmelerine yardımcı olunmaya çalışılacaktır.

CSS NEDİR?



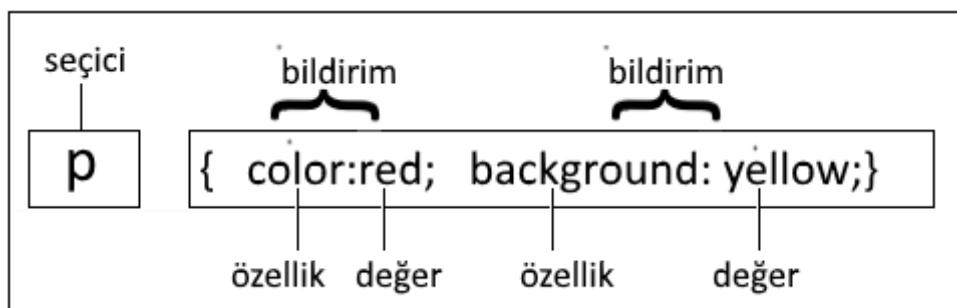
CSS, sayfa stilini ve düzenini tek bir konumda tanımlamaya olanak tanır.

CSS, "Cascading Style Sheets" kelimesinin baş harflerinden oluşmaktadır. CSS (Türkçedeki karşılığı Basamaklı Stil Sayfaları), tek tek sayfalar için stil etiketleri (tag) yazma zorluğunu ve karmaşasını azaltmak için yaratılmıştır. CSS, sayfa şekillendirmeleri için kullanılır. CSS, sayfa stilini ve düzenini tek bir konumda tanımlamaya olanak tanır, böylece onlar genel olarak tüm *.html* dosyalarına uygulanabilir.

```
<style type="text/css">
p{color: red;}
</style>
```

CSS Kural Yapısı

Stil sayfası, sayfa öğelerinin nasıl sunulması gerektiğini açıklayan bir veya daha fazla kuraldan oluşur. Her kuralın iki temel kısmı vardır: *seçici* ve *bildirim bloku* (Şekil 4.1.). Kuralın sol tarafında, kuralın uygulanması gereken belgenin bölmelerini seçen seçici vardır. Seçiciler tek başlarına durabilir veya virgülle ayrılmış bir liste olarak gruplandırılabilir. Bu durumda belirlenen kurallar virgülle ayrılan bütün seçici listesi için geçerli olmaktadır. Kuralın sağ tarafında bildirim bloku vardır. Bir bildirim bloku bir veya daha fazla bildirimden oluşabilir. Her bildirim, bir CSS özelliğinin ve bu özelliğin değerinin bir birleşimidir.



Bütün CSS kurallarında iki temel kısım vardır: seçici ve bildirim bloku.

Şekil 5.1. CSS Kural Yapısı

Bildirim bloku daima kümeye parantezi içine alınır. *Bir bildirim bloku birkaç bildirim içerebilir*. Her bildirim bir noktalı virgül (;) sonlandırılır. Her bildirimde bir özellik, özelliğin değeri, özellik ile özelliğin değeri arasında iki nokta üst üste (:) bulunmaktadır. Özelliğin birden fazla değeri olduğu bazı durumlarda, değerler arasında virgül (,) bulunmaktadır.

Yukarıdaki kodu sayfanın head kısmına eklemek, CSS'yi web sayfasında kullanmanıza imkân sağlayacaktır. Tanımlanan bu kodu sayfada kullanmanız için içerikte tanımladığınız <p> etiketini sayfanızda kullanmanız yeterli olacaktır. Tanımladığınız bu stil, eğer <p> etiketini etkileyen başka bir stil tanımı yoksa HTML sayfanızda kullandığınız bütün <p> etiketlerine uygulanacaktır.



CSS sürümleri World Wide Web konsorsiyumunun önerileri ile ortaya çıkmıştır.

CSS SÜRÜMLERİ

CSS1: CSS'nin bu sürümü ilk olarak Aralık 1996'da World Wide Web konsorsiyumunun bir önerisi olarak ortaya çıkmıştır. CSS dili ile HTML etiketlerinin basit görsel biçimlendirme modellerini açıklar.

CSS2: CSS'nin bu sürümü Mayıs 1998'de World Wide Web konsorsiyumunun bir tavsiyesi olarak ortaya çıkmıştır. İşitsel aygıtlar, çıktılar, öge konumlandırma, tablolar ve indirilebilir yazı tipleri gibi ortama özgü stil sayfalarına destek verir. Önceki sürümün üzerine inşa edilmiştir.

CSS3: CSS'nin bu sürümü de Haziran 1999'da World Wide Web konsorsiyumunun bir tavsiyesi olarak ortaya çıkmıştır. Modül olarak bilinen bölünmüş belgelere sahiptir. Her modül, CSS'nin ikinci sürümünde tanımlanan yeni bir uzantı özelliğine sahiptir. Önceki sürümlerin üzerine inşa edilmiştir. CSS3 modülleri, CSS2'nin eski teknik tanımlamalarına ve uzantılarına da sahiptir. CSS'yi kullanarak metnin

rengini, fontların stilini, paragraflar arasındaki boşluğu, sütunların nasıl boyutlandırıldığını ve düzenlendiğini, hangi arka plan görüntülerinin veya renklerin kullanıldığını, sayfa düzeni tasarımlarını ve farklı cihazlar için ekran boyutları ve diğer efekt varyasyonları kontrol edilebilmektedir. Günümüzde CSS4 olarak adlandırılan tek bir standart spesifikasyonu yoktur. Ancak önceki seviye 3 modülün işlevsellliğini temel alan “Görüntü Değerleri”, “Arka Planlar ve Kenarlıklar” veya “Seçiciler” vb. gibi birkaç seviye 4 modülü bulunmaktadır. Bu seviye 4 modülleri toplu olarak CSS seviye 4 olarak adlandırılabilir.



Bireysel Etkinlik

- CSS3 tanımından sonra birçok değişiklik ortayamasına rağmen, genel kabul görmüş bir CSS4 tanımı olmaması ne tür sorunlara yol açabilmektedir?

CSS TANIMLAMA YÖNTEMLERİ

Satır İçi Stil Tanımlama



Satır içi stiller sadece buradaki tanımlanıkları etiket için geçerlidir. Sayfadaki diğer etiketleri etkilemez.

Satır içi stil tanımlama, etiketin kendi içinde yapılır. *Stil bilgisi, stil özelliği yoluyla tek bir öge için belirtilebilir*. Bir başka ifadeyle sadece buradaki etiket için geçerlidir. Sayfadaki diğer etiketleri etkilemez. Sayfanın her yerinde istenildiği kadar tekrarlanabilir. Çok özel durumlarda (sadece bir – iki defa kullanılması gerekecek) kullanılması tavsiye edilir, *sayfalarda tekrarlanan stil tanımlamaları için önerilmez*. Tekrarlanan stil tanımlamaları için belge düzeyinde stil tanımlama ve dış stil sayfaları kullanılmalıdır.

Tablo 5.1.’de iki tane `<p>` etiketine satır içi stil tanımlama yapılmıştır. Aynı zamanda tarayıcıda tanımlamaların nasıl göründüğü de gösterilmiştir. Tanımlamalarda `<p>` etiketi içinde `style` ifadesi ile stil tanımında bulunacağı belirtilmiştir. Eşittir (=) kısmından sonra ise etiket için stil tanımlaması yapılmaktadır. Stil tanımları çift tırnak (“) içinde yapılmaktadır.

İlk `<p>` etiketi stil tanımında metnin renginin yeşil olması bildirilmiştir. Bu tanımda `p` seçici, `color` özellik ve `green` de özelliğin (`color`) değeridir. İkinci `<p>` etiketi stil tanımında metnin renginin kırmızı ve yazı boyutunun 16 punto olması vurgulanmıştır. İkinci satır için tanımda iki tane stil bildirimini bulunmaktadır. Bildirimlerin arasında ise noktalı virgül (;) kullanılmıştır. İkinci tanımda `p` seçici, `color` ilk özellik, `green` ilk özelliğin değeridir, `font-size` ikinci özellik, `20` de ikinci özelliğin değeridir.

Tablo 5.1. Satır İçi CSS Tanımlama

Html Kodu	Kodun Tarayıcıdaki Görüntüsü
<pre><p style="color: green;"> ilk derse hoş geldiniz. </p> <p style="color: red; font-size:20px;"> ilk derse hoş geldiniz. </p></pre>	<p>ilk derse hoş geldiniz.</p> <p>ilk derse hoş geldiniz.</p>

Belge Düzeyinde Stil Tanımlama



Belge düzeyinde yapılan tanımlamalar, belgede kullanılan bütün etiketler için geçerlidir.

Belge düzeyinde stil tanımlama aynı belgenin başlığında (`<head>` etiketinin içinde) yapılır. Stil tanımlamaları genellikle `<head>` etiketinin içinde bulunsa da bu zorunlu bir kullanım şekli değildir. Bazı durumlarda stiller performans nedeniyle bir belgenin sonuna yakın bir yere de eklenebilirler. *Belge düzeyinde yapılan tanımlamalar, belgede kullanılan bütün etiketler için geçerlidir*. Örneğin Tablo 5.2.'de tanımlanan `<p>` etiketi stili, sayfada bütün `<p>` etiketi kullanımına etki etmektedir (Bu ifade, eğer dosyada aynı etiket için satır içi tanımlama veya bu etiketi etkileyen başka tanımlama yapılmamışsa geçerlidir. Daha detaylı bilgi için CSS'de öncelikler konusunu inceleyebilirsiniz).

Tablo 5.2.'de belge düzeyinde stil tanımlama yapılmıştır. Dosyanın `<head>` etiketi içinde `<style>` etiketi ile başlayıp `</style>` etiketi ile biten bölümün arasında `<h3>` etiketi ve `<p>` etiketi stil tanımlamaları yapılmıştır. `<h3>` etiketi için color (metin rengi) ve background (metin arka plan rengi) bildirimleri, `<p>` etiketi için font-size (metin boyutu) ve text-align (metin hizalama) bildirimleri yapılmıştır.

Dosyanın `<body>` etiketi içinde `<h3>` etiketi ve `<p>` etiketi kullanılmıştır. Bu etiketler her kullanıldığından dosya içinde tanımlanan stiller otomatik olarak bu etiketlere uygulanmaktadır. Kullanıcının başka herhangi bir işlem yapmasına gerek yoktur, tanımlanan etiketleri kullanması tanımlanan stillerin uygulanması için yeterlidir. Dosyanın tarayıcıda açıldığı esnадaki görüntüsü de şeke eklenmiştir.

Tablo 5.2. Belge Düzeyinde Stil Tanımlama

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><html> <head><title>Stiller</title> <style type="text/css"> h3{color:red; background:yellow;} p{font-size:10; text-align:left;} </style> </head> <body> <h3>h3 başlık etiketi </h3> <p> Paragraf etiketi </p> </body></html></pre>	<p>h3 başlık etiketi</p> <p>Paragraf etiketi</p>

Dış Stil Sayfaları

Stiller ayrı bir dosyada saklanabilir. Dış stil sayfalarında, stil bildirimlerini içeren farklı bir dosya oluşturulur ve bunun ardından ana html dosyasına bağlanır. Bu oluşturulan yeni belge “`.css`” uzantısı kullanılarak kaydedilmelidir. Örnek olarak geliştirilen “`still.css`” dosyasını çalışılan dosyaya bağlamak için Şekil 4.2.’deki gibi `<link>` etiketi kullanılmaktadır.



Oluşturulan dış stil sayfaları “`.css`” uzantısı kullanılarak kaydedilmelidir.

Aynı bir stil dosyası kullanmanın en önemli avantajı, yaygın olarak kullanılan stillerin tek bir dosyada toplandığında, bu stilleri kullanan tüm sayfaların tek bir stil sayfasını düzenleyerek güncellenebilmesidir. *Dış stil sayfalarını kullanan bütün .html dosyalarında bu sayfalardaki tanımlanan stil bildirimleri geçerlidir* (bu ifade eğer dış stil sayfalarını kullanan dosyalarda belge düzeyinde ve/veya satır içi stil bildirimi yapılmamışsa geçerlidir. Daha detaylı bilgi için CSS’de öncelikler konusunu inceleyebilirsiniz).

Şekil 4.2.’de `still.css` dosyası içeriği gösterilmektedir. Dosyada `<h3>` etiketi ve `<p>` etiketi stil tanımlamaları bulunmaktadır. `<h3>` etiketi stil tanımlaması için sadece `color` (renk) bildirimi yapılmıştır. `<p>` etiketi stil tanımlaması için `text-align` (metin hizalama), `font-size` (metin boyutu) ve `text-indent` (metin girintisi) olmak üzere üç tane bildirim yapılmıştır.

HTML dosyasının başlığında ise “`still.css`” adlı dış stil dosyasına `<link>` etiketi ile bağlantı kurulmuştur. HTML dosyasının içerik (`<body>`) bölümünde, “`still.css`” dosyasında stil tanımlaması yapılan `<h3>` ve `<p>` etiketleri ile ilgili hiçbir tanımlama yapılmadan doğrudan kullanılmıştır. HTML dosyası tarayıcıda açıldığında, “`still.css`” dosyasında yapılan stil tanımlamalarının uygulandığı görülmektedir. “`still.css`” dosyası başka dosyalara dâhil edildiğinde de benzer şekilde dosya içinde tanımlanan stiller aynı şekilde kullanılabilecektir.

Dış stil sayfalarının en önemli özelliğinin, bu sayfalarda yapılan değişikliklerin sayfaları kullanan HTML sayfalarına otomatik olarak yansıyacağı olduğu söylemiştir. HTML dosyasında herhangi bir değişiklik yapılmadan, “`still.css`” sayfasında `<h3>` etiketi stil tanımlamasında `color` (renk) ve `font-family` (font ailesi) bildirimleri, `<p>` etiketi stil tanımlamasında `text-align` (metin hizalama) ve `text-decoration` (metin dekorasyonu) bildirimleri yapılmıştır. HTML dosyası tarayıcıda açıldığında, “`still.css`” dosyasında yapılan stil değişiklerinin HTML dosyasına otomatik olarak uygulandığı görülmektedir.

still.css dosyası

```

1  h3{color:red;}
2  p{text-align : left; font-size=14; text-indent:20;}
3

```

html sayfası görüntüsü

```

1  - <html><head><title>Stiller</title>
2  <link rel="stylesheet" type="text/css" href="still.css">
3  </head>
4  - <body>
5  <h3>h3 başlık etiketi </h3>
6  <p> Paragraf etiketi </p>
7  </body>
8  </html>X

```

web sayfası görüntüsü

h3 başlık etiketi
 Paragraf etiketi

Şekil 5.2. Dış Stil Sayfaları

CSS SEÇİCİLERİ

Basit Seçici



Basit seçici, etiketin seçici olarak kullanıldığı temel stil tanımlama türündür.

Basit seçici, etiketin seçici olarak kullanıldığı temel stil tanımlama türündür. Tablo 5.3.'teki örnekte olduğu gibi virgül kullanılarak bir stil tanımında çoklu etiket de kullanılabilir. Bu tanımlama ile `<p>` ve `<h1>` etiketlerinin color (renk) özelliklerine yeşil değeri tanımlanmış oldu. *Tanımlanan etiketler HTML dosyasının içine normal etiket kullanımı şeklinde eklenmektedir.* HTML dosyası tarayıcıda açıldığında, yapılan stil tanımlamalarının her iki etikette de uygulandığı görülmektedir.

Tablo 5.3. Basit Seçici Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre> <html> <head><title>Stiller</title> <style type="text/css" > p, h1 {color: blue;} </style></head> <body> <h1>h1 başlık etiketi </h1> <p> Paragraf etiketi </p> </pre>	h1 başlık etiketi Paragraf etiketi

</body> </html>	
-----------------	--

Class (sınıf) seçicisi

Sınıf seçicisi, aynı etiketin iki veya daha fazla tekrarına farklı stiller uygulamak gerekiğinde kullanılabilir. Bu kullanım, etiketteki *class* niteliği bildirerek yapılabilir. Etikette class tanımlamak için etiketin sonuna nokta konup noktadan sonra class adı yazılmalıdır.



Sınıf seçicisi, aynı etiketin iki veya daha fazla tekrarına farklı stiller uygulamak gerekiğinde kullanılabilir.

Sınıf seçicisiyle nasıl bildirim yapıldığı ve bu bildirimlerin kullanılma şekli Tablo 5.4.'te gösterilmektedir. Önce, *<p>* etiketi için "bir" adında bir class oluşturulmuş ve color özelliğinin değeri green yapılmıştır. Sonra, yine *<p>* etiketi için "iki" adında bir class oluşturulmuş ve color özelliğinin değeri red yapılmıştır.

Tablo 5.4. Class (sınıf) Seçicisi Kullanımı

Html Sayfası

```
<html>
<head><title>Stiller</title>
<style type="text/css" >
    p.bir{color:green;}
    p.iki {color:red;}
</style>
</head>
<body>
    <p class="bir">p etiketinin class bir isimli stille kullanımı </p>
    <p class="iki">p etiketinin class iki isimli stille kullanımı </p>
</body>
</html>
```

p etiketinin class bir isimli stille kullanımı

p etiketinin class iki isimli stille kullanımı

Şekil 5.3. Class (sınıf) Seçicisi Kullanımı

Şekil 4.3.'te bildirimlerin tarayıcıda nasıl göründüğü de gösterilmektedir. HTML dosyası içinde tanımlanan class ifadelerini kullanmak için *<p>* etiketi içinde class yazılmış, class ismi de eşittir'den (=) sonra çift tırnak işaretü ("") içinde belirtilmiştir. HTML dosyası tarayıcıda açıldığında ise iki farklı *<p>* class etiketinin ekranda nasıl göründüğü gösterilmektedir.

```
<style type="text/css" >
    p.bir{color:green;}
    p.iki {color:red;}
</style>
```



Bir etiket ile ilgili olarak bir class tanımlandığında, tanımlanan class, etiket için belirlenen bütün stil özelliklerini içerir.

Class kullanımında öncelikle bir etiket stili tanımlayıp aynı etiket ile ilgili olarak bir class tanımlandığında, tanımlanan class, etiket için belirlenen bütün stil özelliklerini içerir. Aynı zamanda *kendisi için belirlenen özellikler* de kullanır. Tablo 5.5.'teki örnekte `<p>` etiketi için color, background, font-size ve font-weight (metin ağırlığı) özellikleri için stil bildirimleri yapılmıştır. Aynı zamanda `<p>` etiketi için "iki" adında bir class oluşturulmuştur. "iki" class tanımlamasında font-size özelliğinin değeri `<p>` etiketi için belirlenen stil değerinden farklı olarak belirlenmiştir. Sayfa tarayıcıda açıldığında (Şekil 5.4.), "iki" class'ı kullandığımız bölümde font-weight haricinde `<p>` etiketinde tanımlanan bütün stillerin aynen kullanıldığı görülmektedir.

Tablo 5.5. Etiket İçin Class ve Stil Tanımlaması

Html Sayfası
<pre> <html> <head><title>Stiller</title> <style type="text/css" > p {color:green; background: yellow; font-size: 12px; font-weight: 800;} p.iki {font-size: 16px;} </style> </head> <body> <p>p etiketinin kullanımı</p> <p class="iki">p etiketinin class iki isimli stille kullanımı</p> </body> </html> </pre> <div style="background-color: #ffff00; padding: 5px; margin-top: 10px;"> p etiketinin kullanımı </div> <div style="background-color: #ffff00; padding: 5px; margin-top: 10px;"> p etiketinin class iki isimli stille kullanımı </div>

Şekil 5.4. Etiket İçin Class ve Stil Tanımlaması

ID Seçici



"id" seçimciler sadece tek bir etikette kullanılabilirler.

ID (kimlik) özelliği ile de stil tanımlamaları yaratılabilir. Class'tan farklı yanları:

- Sadece tek bir etikette kullanılabilirler.
- Aynı "id" değeri iki etikete verilemez (Her "id" sadece tek bir etikette kullanılabilir).
- Stil tanımında *class için .* (nokta) kullanılır, ancak *ID tanımında ise #* (diyez) kullanılır.

Tablo 5.6.'da HTML dosyasının başlığında "bicim" adında bir ID stili tanımlanıyor. Bu tanımda color, background, font-size ve font-weight özellikleri için bildirim yapılıyor. Tablo 5.6.'da aynı zamanda tanımlanan ID stilinin HTML

dosyasında `<p>` etiketi içinde kullanımını da gösterilmektedir. `<p>` etiketinin içinde tanımlanan ID kullanılırken önce “`id`” ifadesi yazılır, sonra eşittir (=) karakterinden sonra çift tırnak içinde belirlediğimiz ID adı (bicim) yazılır. Tablo 5.6.’da HTML dosyasının tarayıcıda açıldığındaki görüntüsü de verilmiştir.

Tablo 5.6. ID (Kimlik) Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><html> <head><title>Stiller</title> <style type="text/css" > #bicim {color:yellow; background: blue; font-size: 16px; font-weight: 800;} </style> </head> <body> <p id="bicim" >ID Kullanimi </p> </body> </html></pre>	ID Kullanimi

CSS VE METİN

Bu bölümde metin düzenleme ile ilgili özellikler incelenecaktır, bu konu ile ilgili çok fazla özellik olduğu için bütün CSS özellikleri kapsamlı bir şekilde incelenmeyecektir. Bunun yerine, metin gösterimi ile ilgili çok sık kullanılan özelliklerden detaylıca bahsedilecektir.

Font (Yazı Tipi)

CSS kullanılarak html belgelerindeki metnin varsayılan yazı tipleri, görünümü ve biçimini değiştirebilir. Yazı tipleri web sayfalarının en temel özelliklerinden biridir. Yazı tipleri, web tasarımının ve Web'de metinlerin nasıl gösterildiğinin ayrılmaz bir parçasıdır. Yazı tipleri, metinleri tekdüzelikten çıkarmak ve farklı ortamlara uygun görüntü ve biçim sağlamak için geliştirilen *semboller ve sayılar topluluğu* olarak tanımlanabilir. Ayrıca, yazı tipleri *belirli bir stilde ve boyutta yazdırılabilen veya görüntülenebilen metin karakterleri topluluğu* olarak da ifade edilebilir. Yazı tipleri sayesinde içerik ile uyumlu metin görüntüleri elde edilebilir. Örneğin, resmî içeriği olan bir web sayfasında Comic MS Sans yazı tipi yerine Arial yazı tipi kullanmak daha uygun olacaktır.

Öncelikle farklı yazı tiplerinin incelenmesi gereklidir. Bütün fontların altına toplandığı font aileleri vardır, bunlar Serif, Sans-serif ve Monospace olmak üzere üç adettir. Serif fontların gözle daha kolay takip edilip okunabildiği, ancak Sans-serif fontların daha modern ve resmî bir görünüme sahip olduğu birçok kişi tarafından kabul edilmektedir.

- *Sans Serif*: Başı ve sonu düz biten, uçlarında uzantısı / çıkıştı olmayan fontlara verilen addır. Örnekler: Calibri, Arial, Verdana.



Yazı tipleri, metinleri tekdüzelikten çıkarmak ve farklı ortamlara uygun görüntü ve biçim sağlamak için geliştirilen semboller ve sayılar topluluğudur.

- **Serif:** Sonunda okumayı kolaylaştırdığı iddia edilen küçük tırnaklar / uzantılar bulunan fontlardır. Örnekler: Times New Roman, Georgia.
- **Monospace:** Tüm alfabe için aynı miktarda boşluk kullanan fontlardır. Bu font grubunda bir cümle içerisinde her harfin kapladığı alan eşittir. Örnekler: Courier New, Lucida Console.

Font-Family (Yazı Tipi Ailesi)

Metnin fontunu değiştirmek için font-family özelliği kullanılır. **font-family** özelliği bildiriminde virgül ayırıcısı kullanılarak birden çok yazı tipi bildirilebilir. Bunun sebebi eğer tarayıcı belirtlen bir yazı tipini desteklemiyorsa, bildirimdeki bir sonraki yazı tipi tarayıcıyla uyumluluk açısından denetlenir. Bildirimde önce en çok istenilen yazı tipi belirtilmeli, sonra ikinci en çok istenen yazı tipi bildirilerek daha genel bir yazı tipine doğru ilerlenmeli, bu şekilde en sonda yazı tipi ailesinden bir yazı tipini seçebilecek şekilde ilan edilmelidir.



font-family özelliği bildiriminde virgül ayırıcısı kullanılarak birden çok yazı tipi bildirilebilir.

Tablo 5.7.'de HTML dosyasında <p> etiketi için iki tane stil tanımaması yapılmıştır. Adı "uzantılı" olan class tanımlamasında font-family ve color özelliği için bildirim yapılmıştır. font-family özelliği için yapılan bildirimde serif yazı tiplerinden üç tanesi kullanılmıştır. Bu ifade ile eğer tarayıcı destekliyorsa, metinler "Droid Serif" fontu kullanılarak tarayıcıda gösterilecek, "Droid Serif" tarayıcı tarafından desteklenmiyorsa metinler "Liberation Serif" fontu kullanılarak tarayıcıda gösterilecek, "Liberation Serif" fontu da tarayıcı tarafından desteklenmiyorsa metinler "serif" fontu kullanılarak tarayıcıda gösterilecek demektir. Benzer şekilde "uzantısız" class kullanılan <p> etiketlerinde öncelikle metinler "Arial", "Arial" yoksa "Calibri", "Calibri" de yoksa tarayıcıda sans-serif bir yazı tipi kullanılarak gösterilecektir. Tablo 5.7.'de her iki class tanımının kullanıldığı HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir.

Yazı tipi adı bir kelimedenden fazlaysa, bu yazı tiplerini bildirirken **çift ters virgül veya çift tırnak** içinde açıklanmalıdır. Örneğin, "Droid Serif" yazı tipinde olduğu gibi yazı tipi ismi çift tırnak arasında yazılmıştır.

```
<style type="text/css">
    p.uzantili{font-family:'Droid Serif','Liberation Serif',serif;
                color:red;}
    p.uzantisiz{font-family:Arial, Calibri,sans-serif;
                text-decoration:underline;}
</style>
```

Tablo 5.7. Font-Family Kullanımı

Html Sayfası	Sayfanın Tarayıcındaki Görüntüsü
<html> <head><title>Stiller</title> <style type="text/css" > p.uzantili {font-family: 'Droid Serif','Liberation	<p>p etiketi Serif (uzantılı)</p> <p>p etiketi Sans-serif (uzantısız)</p>

```

        Serif",serif; color:red;}
    p.uzantisiz {font-
family: Arial, Calibri, sans-serif;
text-decoration:underline;}
</style>
</head>
<body>
<p class="uzantili">p etiketi
Serif (uzantili)</p>
<p class="uzantisiz"> p etiketi
Sans-serif (uzantisiz)</p>
</body>
</html>

```

Font-Size (Yazı Tipi Boyutu)

Yazı tipi boyutunu tanımlamak için *font-size* özelliği kullanılır. Metin boyutları, piksel (px), punto (pt) veya göreceli boyutlardan “em” veya yüzde olmak üzere dört şekilde tanımlanabilir. Varsayılan olarak, tarayıcılarda kullanılan yazı tipi boyutu `<body>` etiketinde tanımlanmıştır ve 16 pikseldir.



Varsayılan olarak, tarayıcılarda kullanılan yazı tipi boyutu `<body>` etiketinde tanımlanmıştır ve 16 pikseldir.

Göreceli boyutlardan “em” ve yüzde, tarayıcının varsayılan metin boyutunun değiştirildiği durumlarda işe yarar. Örneğin em ölçü birimi kullanılarak tanımlanan bir web sayfasında varsayılan yazı tipi boyutu değiştirilirse, sayfadaki bütün metinler otomatik olarak aynı oranda değişecektir.

Tablo 5.8.’de font-size özelliğinin alabileceği değerlerin tanımlanması için `<p>` etiketi ve `<p>` etiketi için dört farklı class stili tanımlaması yapılmıştır. `<p>` etiket kullanımında hiçbir stil tanımlaması yapılmamıştır, bu yüzden tarayıcıda tanımlı font-size boyutunda (`<body>` etiketinin varsayılan metin boyutu) metnin gösterilmesi istenmiştir. Tanımlanan class stillerinde font-size değeri olarak pixel, punto, em ve yüzde değerleri bildirimi yapılmıştır. Tablo 5.8.’de `<p>` etiketi ve `<p>` etiketi için tanımlanan class stillerinin kullanıldığı HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir.

Tablo 5.8. Font-Size Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre> <html> <head><title>Stiller</title> <style type="text/css" > p.pixl {font-size:16px;} p.punt {font-size:16pt;} p.yuzde{font-size:200%;} p.eem{font-size:1.5em;} body{font-size:16px;} </style> </head> <body> <p> p etiketi </p> <p class="pixl">p etiketi 16px</p> <p class="punt">p etiketi 16pt</p> <p class="yuzde">p etiketi 200%</p> <p class="eem">p etiketi 1.5em</p> </body> </html></pre>	<p>p etiketi</p> <p>p etiketi 16px</p> <p>p etiketi 16pt</p> <p>p etiketi 200%</p> <p>p etiketi 1.5em</p>

Tablo 5.9.'da, Tablo 5.8.'de yapılan stil tanımlarında başka hiçbir değişiklik yapılmadan belge düzeyinde stil tanımlamaya uygun olarak `<body>` etiketi için font-size değeri olarak 24 pixel bildirimi yapılmıştır. Tablo 5.9.'da ayrıca bu dosyanın tarayıcıda açılmış hâli de gösterilmektedir. Dosyaların tarayıcıda açılmış hâlleri karşılaştırıldığında pixel ve punto boyutu tanımlanan içeriklerin boyutunun değişmediği; em, yüzde ve `<body>` etiketinin varsayılan değerini kullanan `<p>` içeriklerin boyutunun göreceli olarak değiştiği gözlenmektedir.

Tablo 5.9. Font-Size Varsayılan Değer Etkisi

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre> <html> <head><title>Stiller</title> <style type="text/css" > p.pixl {font-size:16px;} p.punt {font-size:16pt;} p.yuzde{font-size:200%;} p.eem{font-size:1.5em;} body{font-size:24px;} </style></head> <body> <p> p etiketi </p> <p class="pixl">p etiketi 16px</p> <p class="punt">p etiketi 16pt</p> <p class="yuzde">p etiketi 200%</p> <p class="eem">p etiketi 1.5em</p> </body> </html></pre>	<p>p etiketi</p> <p>p etiketi 16px</p> <p>p etiketi 16pt</p> <p>p etiketi 200%</p> <p>p etiketi 1.5em</p>

```
<p class="yuzde">p etiketi  
200%</p>  
<p class="eem">p etiketi  
1.5em</p>  
</body> </html>
```

Text-Align (Metin Hizalama)

Metni sağa, sola veya merkeze hizalamak için text-align özelliği kullanılır. text-align özelliği ile hangi yönün metnin başlangıcı olarak kabul edileceğini ve hangisinin bitiş olarak kabul edileceği belirlenir. Sola (left), merkezde (center), sağa (right) ve iki yana yasla olmak üzere dört seçenekten biri kullanılabilir.



Sola (left), merkezde (center), sağa (right) ve iki yana yasla olmak üzere dört çeşit text-align seçeneği bulunmaktadır.

Tablo 5.10.'da text-align özelliğinin alabileceği değerlerin tanımlanması, `<h3>` etiketinin altında class stilleri tanımlanarak gösterilmiştir. Oluşturulan text-align içeren `<h3>` class tanımlarının HTML dosyasının içerik kısmında kullanımı örnek olarak uygulanmıştır. Şekil 5.5.'te text-align değerlerinin kullanıldığı HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir.

Tablo 5.10. Text-Align Web Sayfası Görüntüsü

Html Sayfası

```
<html>  
<head><title>Stiller</title>  
    <style type="text/css" >  
        h3.sol {text-align:left;}  
        h3.sag {text-align:right;}  
        h3.ortala{text-align:center;}  
        h3.yasla {text-align:justify;}  
    </style>  
</head>  
<body>  
    <h3 class="sol">h3 etiketi sola dayalı</h3>  
    <h3 class="sag">h3 etiketi sağa dayalı</h3>  
    <h3 class="ortala">h3 etiketi ortalanmış</h3>  
    <h3 class="yasla">h3 etiketi iki yana yaslı, h3 etiketi iki yana yaslı,h3 etiketi iki yana yaslı </h3>  
</body>  
</html>
```

h3 etiketi sola dayali**h3 etiketi saga dayali****h3 etiketi ortalanmis****h3 etiketi iki yana yasli, h3 etiketi iki yana yasli, h3 etiketi iki yana yasli**

Şekil 5.5. Text-Align Web Sayfası Görüntüsü

Text-Decoration (Metin Dekorasyon)



Metin dekorasyon özelliği, metnin altından çizgi çizmek, metnin üstünden çizgi çizmek ve metnin üzerinden çizgi çizmek için kullanılır.

Metin dekorasyon özelliği, metnin altından çizgi çizmek (underline), metnin üstünden çizgi çizmek (overline) ve metnin üzerinden çizgi çizmek (line-through) için kullanılabilir. Tablo 5.11.'de text-decoration özelliğinin alabileceği değerlerin tanımlanması `<h3>` etiketinin altında class stilleri tanımlanarak gösterilmiştir. Oluşturulan text-decoration içeren `<h3>` class tanımlarının HTML dosyasının içerik kısmında kullanımı örnek olarak uygulanmıştır. Şekil 5.6.'da text-decoration değerlerinin kullanıldığı HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir.

Tablo 5.11. Text-Decoration Kullanımı

Html Sayfası

```

<html>
<head><title>Stiller</title>
<style type="text/css" >
    h3.ust {text-decoration:overline;}
    h3.alt {text-decoration:underline;}
    h3.uzeri {text-decoration:line-through;}
</style>
</head>
<body>
<h3 class="ust">h3 etiketi çizgi metnin ustunde (overline)</h3>
<h3 class="alt">h3 etiketi çizgi metnin altında (underline)</h3>
<h3 class="uzeri">h3 etiketi çizgi metnin üzerinde (line-through)</h3>
</body>
</html>

```

h3 etiketi çizgi metnin üstünde (overline)

h3 etiketi çizgi metnin altında (underline)

h3 etiketi çizgi metnin üzerinde (line-through)

Şekil 5.6. Text-Decoration Kullanımı

Font-Weight (Metin Koyuluğu)



Font-weight özelliği, metnin ağırlığını (koyuluğunu) tanımlamak için kullanılabilir. Bu özelliği kullanılarak metin normal veya koyu olarak tanımlanabilir.

font-weight özelliği, metnin ağırlığını (koyuluğunu) tanımlamak için kullanılabilir. Bu özelliği kullanılarak metin normal veya koyu olarak tanımlanabilir. font-weight bildiriminde normal, bold (koyu), bolder (daha koyu) ve lighter (ince) değerleri kullanılabilir. Ayrıca 100 (en ince) rakamından başlayıp 100'er artarak 900 (en koyu) sayısına kadar sayılar da değer olarak kullanılabilir. Bu değerlerden 400 normal değerine, 700 de koyu değerine karşılık gelmektedir.

Tablo 5.12.'de satır içi bildirim kullanılarak `<p>` etiketi için font-weight değerleri bildirilmiştir. Stil tanımlamalarında eş değer sayısal ve metin değerleri (bold ile 700, normal ile 400, lighter ile 100) ardı ardına bildirilmiştir. Tablo 5.12.'de ayrıca HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir.

Tablo 5.12. Font-Weight Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre> <html> <head><title>Stiller</title> </head> <body> <p style="font-weight: bold;"> font-weight bold </p> <p style="font-weight: 700;"> font-weight 700 </p> <p style="font-weight: normal;"> font-weight normal </p> <p style="font-weight: 400;"> font-weight 400 </p> <p style="font-weight: lighter;"> font-weight lighter </p> <p style="font-weight: 100;"> font-weight 100 </p> </body> </html> </pre>	font-weight bold font-weight 700 font-weight normal font-weight 400 font-weight lighter font-weight 100

Font-Style (Yazı Tipi Özelliği)



Yazı tipi stili özelliği, yazı tipi stilini normalden italik ya da eğik olarak değiştirmek için kullanılır. font-style stili bildiriminde normal, italic (italik) ve oblique (eğik) değerleri kullanılabilir.

Yazı tipi stili özelliği, yazı tipi stilini normalden italik ya da eğik olarak değiştirmek için kullanılır. font-style stili bildiriminde normal, italic (italik) ve oblique (eğik) değerleri kullanılabilir.

Tablo 5.13.'te satır içi stil bildirimi kullanılarak <p> etiketi için font-style değerleri bildirilmiştir. Stil tanımlamalarında italic, normal ve oblique değerleri kullanılmıştır. Tablo 5.13.'te ayrıca HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir.

Tablo 5.13. Font-Style Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><html> <head><title>Stiller</title> </head> <body> <p style="font-style: italic; font-size: 20px;"> font-style italic </p> <p style="font-style: normal; font-size: 20px;"> font-style normal </p> <p style="font-style: oblique; font-size: 20px;"> font-style oblique </p> </body> </html></pre>	<p><i>font-style italic</i></p> <p>font-style normal</p> <p><i>font-style oblique</i></p>

Text-Transform (Metin Dönüştürme)



text-transform özelliği, metni büyük veya küçük harflere dönüştürmek veya kelimelerin ilk harflerini büyük harfe dönüştürmek için kullanılır. text-transform stili bildiriminde none (değişiklik yapılmaz), lowercase (bütün karakterleri küçük harfe dönüştür), capitalize (bütün kelimelerin ilk harfini büyük harfe dönüştür) ve uppercase (bütün karakterleri büyük harfe dönüştür) değerleri kullanılabilir.

text-transform özelliği, metni büyük veya küçük harflere dönüştürmek veya kelimelerin ilk harflerini büyük harfe dönüştürmek için kullanılır. text-transform stili bildiriminde none (değişiklik yapılmaz), lowercase (bütün karakterleri küçük harfe dönüştür), capitalize (bütün kelimelerin ilk harfini büyük harfe dönüştür) ve uppercase (bütün karakterleri büyük harfe dönüştür) değerleri kullanılabilir.

Tablo 5.14.'te satır içi stil bildirimi kullanılarak <p> etiketi için text-transform değerleri bildirilmiştir. Stil tanımlamalarında none, uppercase, lowercase ve capitalize değerleri kullanılmıştır. Şekil 5.7.'de ayrıca HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir.

Tablo 5.14. Text-transform Kullanımı

Html Sayfası
<pre><html> <head><title>Stiller</title> </head> <body></pre>

```
<p style="text-transform: none;"> paragraf deneme icindir (none) </p>
<p style="text-transform: uppercase;"> paragraf deneme icindir (uppercase)
</p>
<p style="text-transform: lowercase;"> paragraf deneme icindir (lowercase)
</p>
<p style="text-transform: capitalize;"> paragraf deneme icindir (capitalize) </p>
</body>
</html>
```

paragraf deneme icindir (none)

PARAGRAF DENEME ICINDIR (UPPERCASE)

paragraf deneme icindir (lowercase)

Paragraf Deneme Icindir (Capitalize)

Şekil 5.7. Text-transform Kullanımı

Line-Height (Satır Arası Yükseklik)



Satırlar arasındaki yüksekliği belirlemek için line-height özelliği kullanılır.

Satırlar arasındaki yüksekliği belirlemek için line-height özelliği kullanılır. line-height stil bildirimlerinde normal, pixel, punto, em veya yüzde değerleri kullanılabilir. Tablo 5.15.'te satır içi stil bildirimi kullanılarak <p> etiketi için line-height değerleri bildirilmiştir. Aynı metin (şekilde göründüğünden daha fazla) için line-height değeri olarak ayrı ayrı normal ve 2em değerleri bildirilmiştir. Şekil 5.8.'de HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir.

Tablo 5.15. Line-Height Kullanımı

Html Sayfası

```
<html>
<head><title>Stiller</title>
</head>
<body>
<p style="line-height: normal;"> satir arasi yuksekligi belirlemek icin line-height ozelligi kullanilir. </p>
<p style="line-height: 2em;"> satir arasi yuksekligi belirlemek icin line-height ozelligi kullanilir. </p>
</body>
```

**satır arası yüksekliği belirlemek için
line-height özelliği kullanılır.**

**satır arası yüksekliği belirlemek için
line-height özelliği kullanılır.**

Şekil 5.8. Line-Height Kullanımı

Bağlantı (Link) Kullanımı ve Özellikleri



Varsayılan bağlantı stili değiştirilmek istendiğinde, dikkate alınması gereken üç ek bağlantı (hover, active, visited) durumu vardır.

Linklerin farklı durumları vardır, yani bir web sayfasında onlarla etkileşime girildiğinde duruma adapte olurlar. Varsayılan bağlantı stili değiştirilmek istendiğinde, dikkate alınması gereken üç ek bağlantı durumu vardır:

- **hover:** Fare imleci ile bağlantıya tıklamadan önce bağlantının üstüne getirildiğindeki bildirimler için kullanılır.
- **visited:** Web sayfasındaki daha önce ziyaret edilmiş bağlantıların bildirimleri için kullanılır.
- **active:** Bağlantı tıklanma sürecinde olduğundaki bildirimler için kullanılır. Çok hızlı olabilir, bu özelliğin değeri ancak fare düğmesine basılı tutulursa görülebilir.

Tablo 5.16.'da belge düzeyinde stil bildirimi kullanılarak etiketini düzenlemek için hover, visited ve active özellikleri için bildirimler yapılmıştır. Ek bağlantı özellikleri tanımlanırken a harfinden sonra iki nokta üst üste (:) yazıldıkten sonra ek bağlantı adları yazılmıştır. Ek bağlantı özellikleri bildirimlerinde diğer bildirimlerden bir farklılık yoktur. etiketi tanımında color, font-family, text-decoration ve text-transform için bildirimler yapılmıştır. Hover (fare bağlantının üzerine gelince) ek bağlantı özelliği için metin boyutu varsayılan değerinin 1.5 katı (1.5em) olsun bildirimi yapılmıştır. Active (bağlantı tıklanırken) ek bağlantı özelliği için metindeki bütün karakterlerin büyük harfe dönüştürülmesi bildirimi yapılmıştır. Visited (daha önce ziyaret edilmiş bağlantılar) ek bağlantı özelliği için gri renk bildirimi yapılmıştır. Şekilde 5.9.'da etiketi ve her bir ek bağlantının sayfa tarayıcıda açıldığında görüntüleri de bulunmaktadır.

Tablo 5.16. Bağlantı (Link) Kullanımı

Html Sayfası
<pre><html> <head><title>Stiller</title> <style type="text/css" > a {color: red; font-family: Verdana; text-decoration: underline; text-transform: none; } a:hover {font-size: 1.5em;} a:active {text-transform: uppercase;}</pre>

```
a:visited {color: grey;}  
</style>  
</head>  
<body>  
<a href="https://atauni.edu.tr/">Atatürk Üniversitesi</a> <br />  
<a href="https://ataaof.edu.tr/">Açık öğretim Fakültesi</a>  
</body> </html>
```



Şekil 5.9. Bağlantı (Link) Kullanımı



Özet

•CSS Nedir

•CSS, "Cascading Style Sheets" kelimesinin baş harflerinden oluşmaktadır. CSS (Türkçedeki karşılığı Basamaklı Stil Sayfaları), tek tek sayfalar için stil etiketleri (tag) yazma zorluğunu ve karmaşasını azaltmak için yaratılmıştır. CSS, sayfa şekillendirmeleri için kullanılmaktadır.

•CSS Kural Yapısı

•Stil sayfası, sayfa öğelerinin nasıl sunulması gerektiğini açıklayan bir veya daha fazla kuraldan oluşur. Her kuralın seçici ve bildirim bloku üzere iki temel bölümü vardır.

•CSS SÜRÜMLERİ

•World Wide Web konsorsiyumunun tavsiyesi olarak ilk 1996'da ortaya çıkmıştır. Üç tane sürümü vardır, genel kabul görmüş standart bir CSS4 spesifikasyon yoktur.

•CSS TANIMLAMA YÖNTEMLERİ

•**Satır içi:** Satır içi stil tanımlama, etiketin kendi içinde yapılır. Stil bilgisi, stil özelliği yoluyla tek bir öge için belirtilebilir.

•**Belge düzeyinde:** Belge düzeyinde stil tanımlama, aynı belgenin başlığında yapılır.

•**Dış Stil Sayfaları:** Stiller ayrı bir dosyada saklanabilir. Dış stil sayfalarında, stil bildirimlerini içeren farklı bir dosya oluşturulur ve bunun ardından ana html dosyasına bağlanır.

•CSS SEÇİCİLERİ

•**Basit Seçici:** Basit seçici, etiketin seçici olarak kullanıldığı temel stil tanımlama türüdür.

•**Class seçicisi:** Class seçicisi, aynı etiketin iki veya daha fazla tekrarına farklı stiller uygulamamız gereğinde kullanılır.

•**Genel seçici:** Aynı stil tanımlamaları iki veya daha fazla farklı etikete kullanılmak istendiğinde, genel seçici kullanılabilir.

•**ID Seçici:** Sadece tek bir etikette kullanılabilir ve iki etikete verilemeyen özel seçeneklerdir.

•CSS VE METİN

•**Fonts (Yazı tipi):** CSS kullanılarak html belgelerindeki metnin varsayılan yazı tipleri, görünümü ve biçimini değiştirebilir.

•**Font-family:** Metnin fontunu değiştirmek için font-family özelliği kullanılır. font-family özelliği bildiriminde virgül ayırıcısı kullanılarak birden çok yazı tipi bildirilebilir.

•**Font-Size:** Yazı tipi boyutunu tanımlamak için font-size özelliği kullanılır.

•**Text-Align:** Metni sağa, sola veya merkeze hizalamak için text-align özelliği kullanılır.

•**Text-Decoration:** Metin dekorasyon özelliği, metnin altından çizgi çizmek, metnin üstünden çizgi çizmek ve metnin üzerinden çizgi çizmek için kullanılabilir.

•**Font-Weight:** font-weight özelliği, metnin ağırlığını (koyuluğunu) tanımlamak için kullanılabilir.

•**Font-Style:** Yazı tipi stili özelliği, yazı tipi stilini normalden italic ya da egypt olarak değiştirmek için kullanılır.

•**Text-Transform:** text-transform özelliği, metni büyük veya küçük harflere dönüştürmek veya kelimelerin ilk harflerini büyük harfe dönüştürmek için kullanılır.

•**Line-Height:** Satırlar arasındaki yüksekliği belirlemek için line-height özelliği kullanılır.

•**Bağlantı Kullanımı ve Özellikleri:** Linklerin farklı durumları vardır, yani bir web sayfasında onlarla etkileşime girildiğinde duruma adapte olurlar. Varsayılan bağlantı stili değiştirilmek istendiğinde, dikkate alınması gereken üç ek (hover, active ve visited) bağlantı durumu vardır.

DEĞERLENDİRME SORULARI

1. CSS ile ilgili aşağıdaki ifadelerden hangisi söylenemez?
 - a) "Cascading Style Sheets" kelimesinin baş harflerinden oluşmaktadır.
 - b) Türkçedeki karşılığı Basamaklı Stil Sayfalarıdır.
 - c) Resmî olarak 4 sürümü bulunmaktadır.
 - d) Geçerli sürümler World Wide Web konsorsiyumu tarafından önerilmiştir.
 - e) Tüm .html dosyalarına uygulanabilir.
2. CSS tanımlama ile ilgili aşağıdaki ifadelerden hangisi kabul edilemez?
 - a) Satır içi stil tanımlama, etiketin kendi içinde yapılır.
 - b) Belge düzeyinde stil tanımlama sadece aynı belgenin başlığında yapılır.
 - c) Satır içi stil tanımlama sadece tanımlandığı öğeyi etkiler.
 - d) Bir dosyada kullanılacak stil tanımlamaları ayrı bir dosyada yapılabilir.
 - e) Belge düzeyinde yapılan tanımlamalar, belgede kullanılan bütün etiketler için geçerlidir.
3. CSS tanımlama ile ilgili aşağıdaki ifadelerden hangisi söylenemez?
 - a) Basit seçici, etiketin seçici olarak kullanıldığı temel stil tanımlama türüdür.
 - b) Bir stil tanımı birden çok etiket için yapılabilir.
 - c) Aynı etikete farklı stiller uygulayabilmek için sınıf seçicisi kullanılabilir.
 - d) Bir stil tanımı sayfa içinde birden çok etikete uygulanabilir.
 - e) "id" stil tanımlaması birden çok etikette kullanılabilir.
4. `.metin {font-family: 'Droid Serif','Liberation Serif',serif; color:blue;}` stil tanımı ile aşağıdaki ifadelerden hangisi geçerli değildir?
 - a) Stil tanımını uygulamak için etiketlere class bildirimi yapmak gerekir.
 - b) Stil tanımı birden çok etikete uygulanabilir.
 - c) Bu stil tanıtımı uygulanan bir içerik tarayıcıda son seçenek olarak serif yazı tipinde gösterilir.
 - d) Tarayıcıda 'Droid Serif' yazı tipi yoksa, stil uygulanan içerik tarayıcının varsayılan yazı tipi ile gösterilir.
 - e) Stil uygulanan içeriğin gösteriminde öncelik 'Droid Serif' yazı tipininindir.

5. Link tanımlama ile ilgili aşağıdaki ifadelerden hangisi kabul edilemez?
- a) Link stili tanımlama ile linklerin varsayılan mavi rengi değiştirilemez.
 - b) Hover ek özelliği ile imleç link üzerine geldiği zaman stil tanımı yapılır.
 - c) Active ek özelliği imleç ile linke tıklandığı zaman stil tanımı yapılır.
 - d) Visited ek özelliği ile ziyaret edilmiş linkler için stil tanımı yapılır.
 - e) Link tanımlarında ek özelliklerini kullanabilmek için ek özelliğin önüne ":" eklemek gerekmektedir.

6. Şekildeki stil tanımı uygulanan içerik tarayıcıda açıldığında, aşağıdaki ifadelerden hangisi her zaman doğrudur?

```
<p style="font-size:16px;">bir</p>
<p style="font-size:14pt;">iki</p>
<p style="font-size:200%;">üç</p>
<p style="font-size:1.5em;">dört</p>
```

- a) Tarayıcıda en büyük boyutta "üç" metni görünür.
- b) Tarayıcıda "iki" metni, "dört" metninden küçük görünür.
- c) "dört" metni bütün tarayıcılarda aynı boyutta görünür.
- d) "iki" metni bütün tarayıcılarda aynı boyutta görünür.
- e) "üç" ve "dört" metinlerinin boyutu tarayıcının varsayılan boyutuna göre değişmez.

7. Şekildeki görüntüyü elde etmek için uygulanan stil tanımlaması aşağıdakilerden hangisi olabilir?

Ataturk AOF

- a) `<h3 style="text-transform: lowercase; font-style:italic;">Ataturk AOF</h3>`
- b) `<h3 style="font-style: italic; font-weight:800;">Ataturk AOF</h3>`
- c) `<h3 style="text-decoration: overline; font-style:italic;">Ataturk AOF</h3>`
- d) `<h3 style="text-decoration: line-through; font-style:italic;">Ataturk AOF</h3>`
- e) `<h3 style="text-decoration: line-through; font-style:oblique;">Ataturk AOF</h3>`

8. Şekildeki görüntüyü elde etmek için uygulanan stil tanımlaması aşağıdakilerden hangisi olabilir?

Atatürk Üniversitesi, Açık Öğretim

Fakültesi, Bilgisayar Programcılığı

Atatürk Üniversitesi, Açık Öğretim
Fakültesi, Bilgisayar Programcılığı

- a) `<p style="font-size:2em; background:#8877DD;">
<p style="font-size:1em;">`
- b) `<p style="font-weight:800; background:#8877DD;">
<p style="font-weight:400;">`
- c) `<p style="line-height:2em; background:#8877DD;">
<p style="line-height:1em;">`
- d) `<p style="text-decoration:overline; background:#8877DD;">
<p style="text-decoration:underline;">`
- e) `<p style="text-transform:uppercase; background:#8877DD;">
<p style="text-transform:lowercase;">`
9. Linkin üzerinde geldiğinde linkin rengini yeşil ve metni koyu, linkin üzerine tıklandığında yazı boyutunu iki katına çıkaran ve ziyaret edilmiş linklerin rengini gri yapan stil tanımı aşağıdakilerden hangisidir?
- a) `a:hover{color:green; font-weight:bold;} a:active{font-size:2em;}
a:visited{color:grey;}`
- b) `a:hover{color:green; font-weight:300;} a:active{font-size:2em;}
a:visited{color:grey;}`
- c) `a:hover{color:grey; font-weight:300;} a:active{font-size:2em;}
a:visited{color:green;}`
- d) `a:hover{color:grey; font-weight:bold;} a:active{font-size:2em;}
a:visited{color:green;}`
- e) `a:hover {font-size:2em;} a:active {color:green; font-weight:bold;}
a:visited{color:grey;}`

10. Aşağıdaki stil tanımlarından hangisi şekillerdeki görüntünün elde edilmesini sağlar? Sayfa açıldığında sayfada iki tane link (Şekil a) görünmektedir. İmleç ilk linkin üzerine geldiğinde sayfa Şekil b'deki gibi görünmektedir.

Ataturk AOF
Açık Öğretim AOF

Şekil a

ATATURK AOF
Açık Öğretim AOF

Şekil b

- a `<style>a:hover{font-weight:bold; text-decoration:none;}`
- b `<style>a:hover{text-transform:uppercase; text-decoration:none;}`
- c `<style>a:hover{text-transform:capitalize; text-decoration:none;}`
- d `<style>a:hover{text-transform:uppercase; font-style:none;}`
- e `<style>a:hover{text-transform:capitaliz`

Cevap Anahtarı

1.d, 2.b, 3.e, 4.d, 5.a, 6.d, 7.c, 8.c, 9.a. 10.b

YARARLANILAN KAYNAKLAR

- Brown, T. B. (2018). CSS master (2nd Edition). VIC: SitePoint Pty. Ltd.
- Dean, J. (2019). Web programming with HTML5, CSS, and Javascript. MA: Jones & Bartlett Learning, LLC, an Ascend Learning Company.
- Grant, K. J. (2018). CSS in Depth. NY: Manning Publications Co.
- Meyer, E. A. (2018). CSS Pocket Reference (5th Edition). CA: O'Reilly Media, Inc.

CSS'TE RENK TANIMI VE TABLO DÜZENLEME



İÇİNDEKİLER

- HTML5 ile Yeni Veri Giriş (Input) Tipleri
- HTML5 ile Yeni Form Özellikleri
- Form Örnekleri



HEDDEFLER

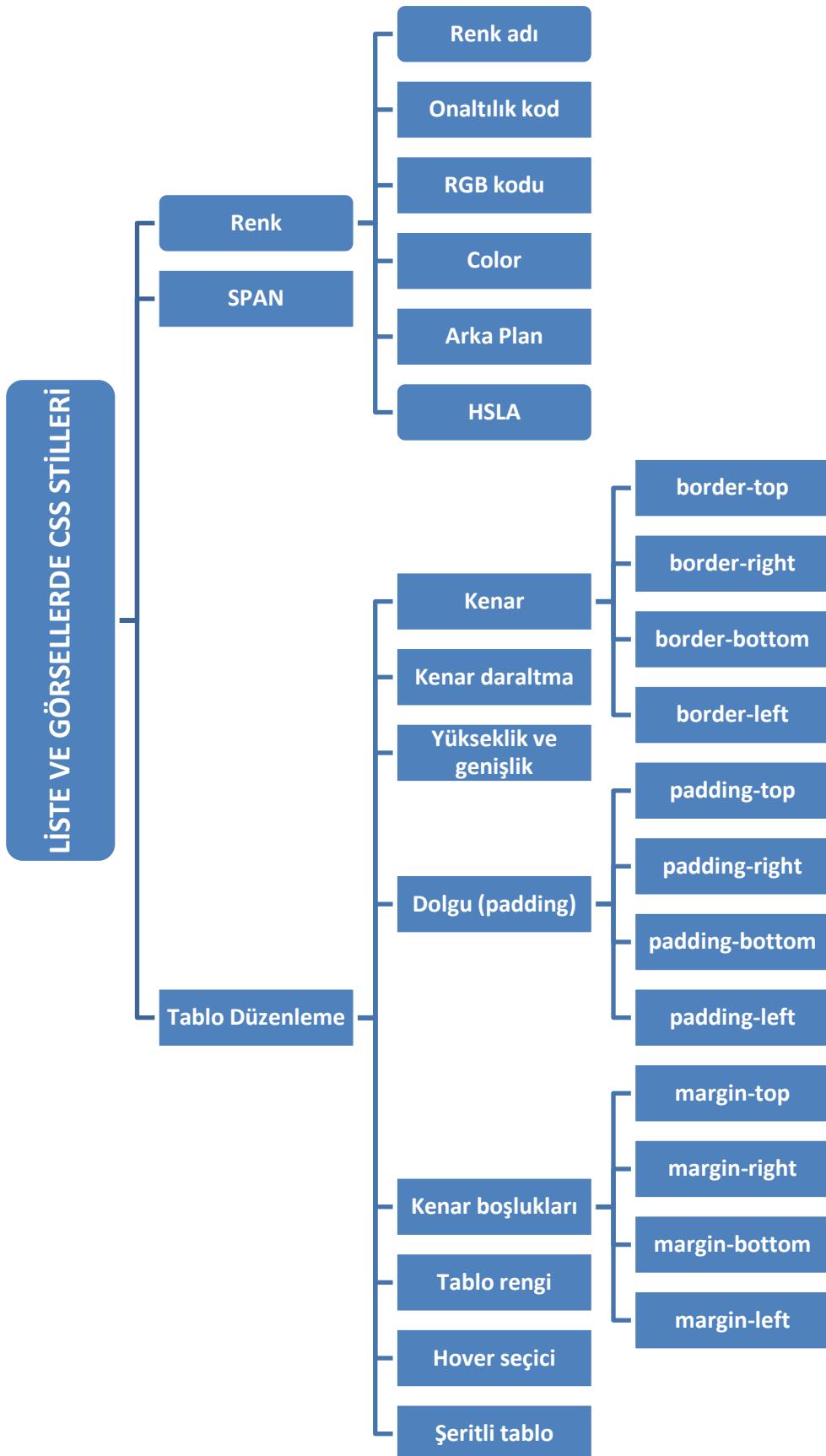
- Bu üniteyi çalıştıktan sonra;
 - CSS tanımında kullanabilecek renk değerlerini açıklayabilecek,
 - CSS tanımında kullanabilecek renk özelliklerini kavrayabilecek,
 - SPAN etiketinin kullanılmasını tanımlayabilecek ve tanımlama sonuçlarını gösterebilecek,
 - Margin kullanımını açıklayabilecek ve margin tanımlarının sonuçlarını sunabilecek,
 - Padding kullanımını açıklayabilecek ve padding tanımlarının sonuçlarını ifade edebilecek,
 - Tablo stillerini tanımlamak için gerekli özellikleri açıklayabilecek,
 - Tablo görüntülerini oluşturan css tanımlarını seçebileceksiniz.



Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET PROGRAMCILIĞI II Doç. Dr. Ercan TOP

ÜNİTE
6



GİRİŞ

Önceki bölümde web sayfalarını şekillendirmek için kullanılan, etiketleri yazma zorluğunu ve karmaşasını azaltan CSS (Cascading Style Sheets- Basamaklı Stil Sayfaları) kavramından bahsedilmiştir. Sonrasında ise stil sayfalarının oluşturduğu kural yapıları, CSS sürümleri, stil tanımlama yöntemleri, stil tanımında kullanılan seçici ve bildirim bloku ve seçici türleri hakkında detaylı bilgiler verilmiştir. Bölümün geri kalanında ise metin düzenlemeyle ilgili sık kullanılan özellikler ve bu özelliklere atanabilecek değerlerden bahsedilmiştir. Metinler ile ilgili yazı tipleri (font), yazı tipi aileleri (font-family), yazı tipi boyutu (font-size), metin hizalama (text-align), metin dekorasyon (text-decoration), metin ağırlığı (font-weight), yazı tipi özelliği (font-style), metin dönüştürme (text-transform), bağlantı kullanımı (link) ve satır yüksekliği (line-height) özellikleri hakkında bilgiler verilip örnekler gösterilmiştir.

Bu bölümde ise öncelikle web sayfalarında renk kavramı hakkında bilgiler verilip; color, background ve HSLA özelliklerinden ve tanımlanma yöntemlerinden bahsedilecektir. Span özelliği ile ilgili bilgiler ve kullanımından örnekler verilecektir. Sonrasında ise tabloların stillerinin düzenlenmesi ile ilgili olarak border, border-collapse, height, width, padding, margin, table color, hover ve nth-child özelliğinden bahsedilecektir. Her bir özellik hakkında önce açıklayıcı bilgiler verilecektir. Sonrasında ise özelliklerin kullanım şekilleri ve yöntemleri hakkında detaylı açıklamalar yapılacak, kullanım şekilleri ve yöntemleri ile ilgili tam HTML dosyaları veya HTML dosyalarının bazı bölümlerinin ekran görüntüleri paylaşılacaktır. Hazırlanan HTML kodlarının tarayıcılarında açıldığında ortaya çıkan görüntüler de HTML dosyalarının içerikleriyle birlikte verilerek, öğrencilerin HTML kodu ve bu kodlarla ortaya çıkacak görüntüyü zihinlerinde eşleştirmelerine yardımcı olunmaya çalışılacaktır.

RENK



Metin rengini ayarlamak için color, arka plan rengini ayarlamak için background-color ve kenarlık rengini ayarlamak için border-color özelliği kullanılmaktadır.

Renkler, ilgili herhangi bir CSS kodu kullanılarak web sitesine veya blog'a uygulanabilirler. Metin rengini ayarlamak için **color**, arka plan rengini ayarlamak için **background-color** ve kenarlık rengini ayarlamak için **border-color** özelliği kullanılmaktadır. Bütün özelliklerde aynı renk bildirim çeşitleri kullanılabilmektedir. Color bildirimi için 3 farklı yöntem vardır.

Renk Adı

Renk adı doğrudan beyan edilerek renk bildirilebilir. **Adlandırılmış renkler İngilizce isimleriyle bildirilmek zorundadırlar ve büyük / küçük harfe duyarlı değillerdir.** Bu nedenle, blue, Blue, BLUE ve bLuE ifadelerinin tümü geçerli ve aynı renk adlarıdır. Tüm modern tarayıcılar adlandırılmış 140 rengi (örneğin, AliceBlue, AntiqueWhite, Aqua, Aquamarine, Azure) destekler. **Bütün adlandırılmış renkler hexadecimal ve RGB kodu ile de bildirilebilirler.**

Onaltılık Kod (Hexadecimal Code)

Renk, her 2 hanenin kırmızı, yeşil, mavi renklerini temsil ettiği 6 basamaklı onaltılık sayı kullanılarak bildirilebilir. Onaltılık kod bayt değerleri, *bir rengin en düşük yoğunluğu olan 00 ile en yüksek yoğunluğu temsil eden FF* arasındadır. Hexadecimal renk kodları bütün tarayıcılar tarafından desteklenmektedir. 24 bit renk yelpazesinin tamamını destekleyen modern tarayıcılarda, *16.777.216 farklı renk* seçeneği bulunmaktadır. Sayı beyanından önce # yazılmalıdır. Örneğin beyaz renk, her üç ana rengin her birinin tam yoğunluğunda karıştırılmasıyla elde edilir ve bu yüzden beyaz rengi tanımlamak için #FFFFFF hexadecimal değer kullanılmalıdır. Ekrandaki herhangi bir rengin olmaması siyah olarak ifade edilmektedir. Siyah rengi bildirmek için her rengin mümkün olan en düşük yoğunluktaki hexadecimal kodunun "00" kullanımıyla #000000 değeri ortaya çıkar. Kırmızı, yeşil ve mavi olan üç ana renk, istenen rengin en yüksek yoğunluğunu diğer ikisinin en düşük yoğunluğuyla karıştırarak yapılır. Örneğin kırmızı rengi elde etmek için kırmızıdan en yüksek yoğunluk değeri olan FF, yeşil ve mavi için de en düşük yoğunluk (hiç) değeri olan 00 kullanılarak #FF0000 hexadecimal kodu elde edilir. Benzer şekilde yeşil ve mavi temel renk kodu da elde edilebilir.

RGB Kodu

İstenilen rengi elde etmek için ne kadar *kırmızı, yeşil ve mavi rengin kullanıldığı bildirilen RGB yöntemi* kullanılarak da renk seçimi yapılabilir. Beyanlar *rgb* ile başlayıp parantez içinde virgülerle ayrılmış üç sayı blokundan oluşmaktadır. Her bir sayı bloğunun alabileceği değerler *0 ile 255* arasındadır (örnek RGB kodu kullanımı *rgb(0, 255, 255)*). RGB genel olarak bu şekilde kullanılmaktadır. Ama CSS3'te 0 ile 1 arasında değer alabilen *opaklık değeri de dördüncü özellik* olarak tanımlanabilmektedir. Örneğin renk bildirimi için *rgb(100, 25, 25, 0.50)* kullanılan tanımda opaklık değeri %50 yani yarı saydamdır. Opaklık değeri 0 olsaydı tam saydam olacaktı. Renk tanımında kullanılan *Hexadecimal sayı değerlerinin hepsi rgb kodu olarak* da ifade edilebilmektedir. Bütün adlandırılmış renkler rgb ve hexadecimal kod olarak da ifade edilebilmektedir.

Renk (Color)

Renk özelliği, metnin ön plan rengini değiştirmek için kullanılır. Bir başka ifade ile metnin rengini değiştirmek için kullanılır. Color özelliği bütün tarayıcıların *ilk sürümünden* itibaren desteklenmektedir.

Tablo 6.1.'de satır içi stil bildirimi kullanılarak <p> etiketi için color değeri olarak blue (mavi) üç farklı şekilde bildirilmiştir. Bildirimlerde renk adı olarak blue, hexadecimal değer olarak #0000FF ve RGB değeri olarak rgb(0, 0, 255) kullanılmıştır. Tablo 6.1.'de ayrıca HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir. Her üç tanımda da aynı şekilde rengin mavi olarak tarayıcıda görüldüğü görülmektedir.



Hexadecimal renklerde renk, her 2 hanenin kırmızı, yeşil, mavi renklerini temsil ettiği 6 basamaklı onaltılık sayı kullanılarak bildirilebilir.



RGB renk bildiriminde beyanlar *rgb* ile başlayıp parantez içinde virgülerle ayrılmış üç sayı blokundan oluşmaktadır.

Tablo 6.1. Color Özelliği Kullanımı

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><html> <head><title>Stiller</title> </head> <body> <p style="color: blue">p etiketi color:blue</p> <p style="color: #0000FF">p etiketi color:#0000FF</p> <p style="color: rgb(0,0,255)">p etiketi color:rgb(0,0,255)</p> </body> </html></pre>	<p>p etiketi color:blue</p> <p>p etiketi color:#0000FF</p> <p>p etiketi color:rgb(0,0,255)</p>

Arka Plan (background)



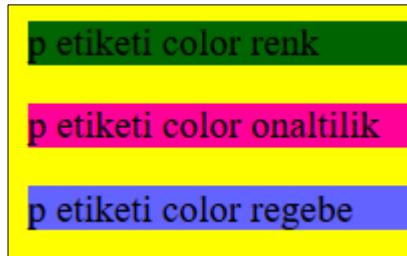
Arka plan renklerini bildirmek için background-color özelliği kullanılır.

Arka plan renklerini bildirmek için **background-color** özelliği kullanılır, RENK bölümündeki renk bildirmek için kullanılan bildirim türleri aynı şekilde burada da kullanılabilir. Tablo 6.2.'de belge düzeyinde stil tanımlama ile <body> etiketi için arka plan renk bildirimi ve <p> etiketi için arka plan renk bildirimi içeren üç tane class stili tanımlaması yapılmıştır. **<body> etiketi için tanımlanan renk, bütün HTML sayfasının arka planına uygulanmaktadır.** Class bildirimlerinde üç farklı renk bildirim yöntemi de kullanılmıştır. Sayfa içerisinde tanımlanan class stilleri ayrı ayrı <p> etiketlerine uygulanmıştır. Şekil 6.1.'de ayrıca HTML dosyasının tarayıcıda açılmış hali gösterilmektedir. Sayfa gösteriminde, sayfa arka plan rengi ve her bir paragrafin arka plan renginin ayrı ayrı olduğu görülmektedir.

Tablo 6.2. Background Özelliği Kullanımı

Html Sayfası
<pre><html> <head><title>Stiller</title> <style type="text/css"> body {background-color: yellow;} p.renk {background-color: DarkGreen;} p.onaltilik{background-color:#ff0099;} p.regebe{background-color: rgb(99,99,256);} </style> </head> <body> <p class="renk">p etiketi color renk</p> <p class="onaltilik">p etiketi color onaltilik</p> <p class="regebe">p etiketi color regebe</p> </body></pre>

```
</html>
```



Şekil 6.1. Background Özelliği Kullanımı

HSLA

HSLA; renk (hue), doygunluk (saturation), ışık miktarı (lightness) ve alfa (saydamlık / opacity) değerleri kullanılarak yapılan yeni nesil renk bildirimidir.

- ***Renk (Hue)***: Renklerin açıları kullanılarak seçildiği renk çemberinin bir açısını temsil eder. Değeri 0 ile 360 arasında değişen derece cinsinden bir açı olarak belirtilir.
- ***Doygunluk (Saturation)***: Renkteki gri miktarıdır, %0 gri bir gölgeyi ve %100 gri renge tam doygunluğun olduğunu belirtmek için kullanılır.
- ***İşik Miktarı (Lightness)***: Luminescence olarak da kullanılmaktadır. Renkteki siyah ve beyaz miktarını belirtmek için kullanılan değerdir. Tamamen siyah %0, normal renk %50 ve tamamen beyaz %100 ile ifade edilmektedir.
- ***Opacity***: Saydamlık değeridir. Renk (color) başlığında detaylı olarak açıklanmıştır.

HSLA; renk (hue), doygunluk (saturation), ışık miktarı (lightness) ve alfa (saydamlık / opacity) değerleri kullanılarak yapılan renk bildirimidir.

Tablo 6.3.'te belge düzeyinde stil tanımlama ile <body> etiketi için arka plan renk bildirimi ve <p> etiketi için arka plan renk bildirimi içeren iki tane class stil tanımlaması yapılmıştır. Bütün üç bildirimde de mavi renk ve farklı alfa (opaklılık) değerleri kullanılmıştır. Tablo 6.3.'te ayrıca HTML dosyasının tarayıcıda açılmış hali de gösterilmektedir. Sayfa gösteriminde sayfa arka plan rengi ve her iki paragrafın arka plan renginin farklı mavi tonlarında olduğu görülmektedir.

Tablo 6.3. HSLA Özelliği Kullanımı

Html Sayfası	Sayfanın Tarayıcıda Görüntüsü
<pre><html> <head><title>Stiller</title> <style type="text/css"> body{background-color: hsla(240, 55%, 35%, 0.1);} p.hsla05{background- color: hsla(240, 55%, 35%, 0.5);} p.hsla1{background-color: hsla(240, 55%, 35%, 1);} </style></head><body></pre>	

```
<p class="hsla05">P etiketi color  
hsla a:0.5</p>  
<p class="hsla1">P etiketi color  
hsla:a 1</p>  
</body> </html>
```

SPAN

`` etiketi, satır içi öğeler ve içerik için genel bir satır içi kapsayıcıdır. Öğeleri stillendirme amacıyla gruplamak için kullanılır (sınıf veya kimlik özniteliklerini kullanarak), başka hiçbir etiket mevcut olmadığından kullanmak için daha iyi bir yoldur. Span etiketi eşleştirilmiş bir etikettir, hem açık (`<`) hem de kapama (`>`) etiketine sahiptir ve etiketi kapatmak zorunludur. Özettemek gerekirse `` etiketi satır içi öğelerin gruplandırılmasında kullanılır ve kendi başına herhangi bir görsel değişiklik yapmaz. Etiket sözdizimi ` içerik metni ` şeklinde yapılmaktadır.

Bir paragrafin içerisinde belirli bir bölüme vurgu /değişiklik (örneğin, font-size, font-family, background-color, color, text-decoration, font-style, text-transform, font-weight vb özellikleri kullanarak) yapmak istendiğinde paragrafin geri kalanını etkilemeden sadece istenilen bölüme uygulamak için `` etiketi kullanılabilmektedir. Tablo 6.4.'te `<p>` etiketinin içindeki "Bilgisayar" kelimesine (`` etiketine) "farklilik" class tanımı uygulanmıştır. "farklilik" class tanımında metin color özelliği için kırmızı, text-decoration özelliği için üzeri çizgili, text-transform özelliği için büyük harf, font-style özelliği için eğik ve background-color özelliği için "#99DDFF" hexadecimal değerleri bildirilmiştir. Şekil 6.2.'deki web sayfası görüntüsü incelemişinde tanımlanan class stilinin sadece "Bilgisayar" kelimesine (`` etiketi ile başlayıp `` etiketi ile biten bölüme) uygulandığı görülmektedir.



Bir paragrafin içerisinde belirli bir bölüme vurgu yapmak istediginde paragrafin geri kalanını etkilemeden sadece istenilen bölüme uygulamak için `` kullanılabilir.

Tablo 6.4. Span Etiketi Kullanımı

Html Sayfası

```
<html>  
<head><title>Stiller</title>  
<style type="text/css">  
.farklilik{color:red; text-decoration: line-through; text-transform:uppercase;  
font-style:italic; border:1px solid black; background-color:#99DDFF;  
padding:10px;}  
</style>  
</head>  
<body>  
<p>Atatürk Üniversitesi, Açık Öğretim Fakültesi, <span class= "farklilik">  
Bilgisayar </span> Programcılığı Bölümü</p>  
</body>  
</html>
```

Şekil 6.2. Span Etiketi Kullanımı

TABLO DÜZENLEME

Border (Kenar)

CSS'de tablo kenarlığını belirtmek için border özelliği kullanılır. Border özelliğinin, **border-width** (kenarlık genişliği), **border-style** (kenarlık stili) ve **border-color** (kenarlık rengi) olmak üzere üç alt özelliği bulunmaktadır. Üç alt özelliğin sadece border özelliği ile tek seferde tanımlamak mümkündür. Border özelliğinin sözdizimi "**border: width style color;**" olarak belirlenmiştir. Sözdiziminde kullanılan width, kenarlığın ağırlığını veya genişliğini belirtir. Sözdiziminde kullanılan style, kenarlık için bir stil (kenarlığın noktalı, kesikli, düz vb.) belirtir. Sözdiziminde kullanılan color ise kenarlığın rengini belirtmektedir. Border özelliğinin alt özelliklerini ayrı ayrı olarak da tanımlamak mümkündür. Border ve border-width özelliği kenarlık genişliğini tanımlamak için kullanılan özelliktir. Bununla birlikte border-style ise kenarlık stilini, border-color ise kenarlık rengini tanımlamak için kullanılan özelliktir.

Border özelliğinde kenarlık stili için aşağıdaki değerlere izin verilmektedir:

- **dotted**- Noktalı kenarlık tanımlar.
- **dashed**- Kesikli kenarlık tanımlar.
- **solid**- Kesintisiz bir kenarlık tanımlar.
- **double**- İkili kenarlık tanımlar.
- **groove**- 3B oluklu kenarlığı tanımlar. Etki, kenarlık rengi değerine bağlıdır.
- **ridge**- 3D sırt çizgisini tanımlar. Etki, kenarlık rengi değerine bağlıdır.
- **inset**- 3B iç metin kenarlığını tanımlar. Etki, kenarlık rengi değerine bağlıdır.
- **outset**- 3B başlangıç sınırını tanımlar. Etki, kenarlık rengi değerine bağlıdır.



Border özelliği için dotted, dashed, solid, double, groove, ridge, inset ve outset değerleri kullanılmaktadır.

Tablo 6.5.'te yer alan örnekteki bildirim `<table>`, `<th>` ve `<td>` ögelerinin siyah kenarlığını (1 pixel kalınlığında ve düz siyah) belirtmek için kullanılmıştır. Sayfanın tarayıcıda açıldığı esnادaki şeklini gösteren Şekil 6.3.'te tablonun çift kenarlığa sahip olduğu görülmektedir. Bunun nedeni hem tablonun hem de `<th>` ve `<td>` ögelerinin ayrı kenarlıklarını olmasıdır.

Sonraki bölümlerde tanımlanan css dosyalarının gösterimi için şekildeki bu tablo kullanılacaktır. Gösterimlerin çok fazla yer kaplamaması için sadece css bildirimleri ve sonuçlarının tarayıcıda gösterimi yapılacaktır. Tablonun HTML kodu ile oluşturulması tekrar gösterilmeyecektir.

Tablo 6.5. Tablo HTML Kodu

Html Sayfası
<pre><html> <head><title>Stiller</title></pre>

```

<style type="text/css">
    table, th, td { border: 1px solid black;}
</style>
</head>
<body>
<table>
    <tr><th>İsim</th><th>Soyisim</th><td>Sancar</td></tr>
    <tr><td>Aziz</td><td>Sancar</td><td>72</td></tr>
    <tr><td>Mert</td><td>Gazoz</td><td>20</td></tr>
    <tr><td>Ali</td><td>Topaloğlu</td><td>19</td></tr>
</table>
</body>
</html>

```

İsim	Soyisim	Yaş
Aziz	Sancar	72
Mert	Gazoz	20
Ali	Topaloğlu	19

Şekil 6.3. HTML Kodunun Web Sayfasında Görünümü

Kenarlık yalnızca tablonun dışına verilmek istenirse, *kenarlık özelliği yalnızca tablo etiketi* için bildirebilir. Tablo 6.6.'da sadece tablonun bütün kenarları için kenarlık değeri (bir piksel, düz ve siyah) atanmaktadır ve tarayıcıda bu bildirim sonucu da gösterilmektedir.

Tablo 6.6. Border (Kenar) Özelliği Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü												
<pre> <style type="text/css"> table { border: 1px solid black;} </style> </pre>	<table border="1"> <thead> <tr> <th>İsim</th><th>Soyisim</th><th>Yaş</th></tr> </thead> <tbody> <tr> <td>Aziz</td><td>Sancar</td><td>72</td></tr> <tr> <td>Mert</td><td>Gazoz</td><td>20</td></tr> <tr> <td>Ali</td><td>Topaloğlu</td><td>19</td></tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

Border özelliği şimdije kadar bütün kenarları tek bir defada tanımlamak için kullanılmıştı. `<table>`, `<th>` ve `<td>` etiketleri için istenilirse *her bir kenar için ayrı ayrı kenarlık* tanımlanabilir. Üst kenarlık için `border-top`, sağ kenarlık için `border-right`, alt kenarlık için `border-bottom` ve sol kenarlık için `border-left` özelliklerini kullanılabılır. Tablo 6.7.'de `<table>` etiketi için `border-bottom`, `<th>` etiketi için `border-top` ve `<td>` etiketi için `border-left` özelliği için beş piksel değerinde farklı renklerde bildirim yapılmıştır. Tarayıcıda bildirim sonuçları görünümü incelendiğinde üst kenarlık için `<th>` etiketi bildirimi olan siyah renk, hücrelerin solundaki kenarlık için `<td>` etiketi bildirimi olan mavi renk ve alt kenarlık için `<table>` etiketi bildirimi olan kırmızı rengin kullanıldığı gözlemlenmektedir.



Border özelliği istenilirse her bir kenar için ayrı ayrı kenarlık tanımlanabilmektedir.

Tablo 6.7. Table Border Özelliği Kullanımı

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü												
<pre><style type="text/css"> table{border-bottom: 5px solid red;} th{border-top: 5px solid black;} td{ border-left:5px solid blue;} </style></pre>	<table border="1"> <thead> <tr> <th>İsim</th> <th>Soyisim</th> <th>Yaş</th> </tr> </thead> <tbody> <tr> <td>Aziz</td> <td>Sancar</td> <td>72</td> </tr> <tr> <td>Mert</td> <td>Gazoz</td> <td>20</td> </tr> <tr> <td>Ali</td> <td>Topaloğlu</td> <td>19</td> </tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

Kenar daraltma (border-collapse)

Tablo kenarlıklarının tek bir kenarlığa daraltılması veya standart HTML'deki gibi mi ayrılacağını belirler. Tablo hücreleri etrafındaki kenarlık ile olan mesafeyi düzenlemeye olanak sağlar. Border-collapse, *collapse* (daralt) ve *separate* (ayırık) değerlerini alabilir. Genelde tarayıcılarda varsayılan değeri *separate* olarak belirlenmiştir. Eğer `<table>` etiketinde border-collapse değeri olarak collapse değeri kullanılırsa bütün tablo etiketlerine kenarlık eklenmesine rağmen Tablo 6.8.'deki gibi tablo hücreleri arasında tek kenarlık görüntülenecektir.

Tablo 6.8. Border-collapse Özelliği Kullanımı

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü												
<pre><style type="text/css"> table, th, td {border-bottom: 1px solid black;} table {border-collapse: collapse;} </style></pre>	<table border="1"> <thead> <tr> <th>İsim</th> <th>Soyisim</th> <th>Sancar</th> </tr> </thead> <tbody> <tr> <td>Aziz</td> <td>Sancar</td> <td>72</td> </tr> <tr> <td>Mert</td> <td>Gazoz</td> <td>20</td> </tr> <tr> <td>Ali</td> <td>Topaloğlu</td> <td>19</td> </tr> </tbody> </table>	İsim	Soyisim	Sancar	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Sancar											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

Yükseklik ve Genişlik (Height ve Width)



Yüksekliği ve genişliği bildirmek için yüzdelik değerler (%), em veya tam değerler (piksel, pt, cm) kullanılabilir.

Tablonun ve tablodaki hücrelerin genişlik ve yüksekliği, height ve width özelliklerinden tanımlanır. Yüksekliği ve genişliği bildirmek *için yüzdelik değerler (%), em* veya *tam değerler (piksel, pt, cm)* kullanılabilir. Aynı zamanda yükseklik ve genişlik otomatik olarak ayarlanabilir (Bu varsayılan değerdir. Tarayıcının yüksekliği ve genişliği hesapladığı anlamına gelmektedir). Tablo 6.9.'daki örnek, tablonun genişliğini 300 piksel, `<th>` ve `<td>` etiketlerinin yüksekliğini 50 piksel, genişliğini de %33 olarak bildirir. Tablonun genişliğini 300 piksel olarak ayarlamak, sayfa tarayıcıda ne kadar büyük açılırsa açılsın tablonun kapsayacağı alan 300 piksel olacak demektir. Border ve border-collapse değerleri ile ilgili yukarıdaki başlıklarda bilgi verilmiştir.

Tablo 6.9. Height ve Width Özelliği Kullanımı

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü												
<pre><style type="text/css"> table{border: 1px solid black; border-collapse:collapse; width:300px; } td, th{height:50px; width:%33; border: 1px solid black;} </style></pre>	<table border="1"> <thead> <tr> <th>İsim</th> <th>Soyisim</th> <th>Sancar</th> </tr> </thead> <tbody> <tr> <td>Aziz</td> <td>Sancar</td> <td>72</td> </tr> <tr> <td>Mert</td> <td>Gazoz</td> <td>20</td> </tr> <tr> <td>Ali</td> <td>Topaloğlu</td> <td>19</td> </tr> </tbody> </table>	İsim	Soyisim	Sancar	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Sancar											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

Dolgu (padding)

Padding özelliği, `<td>` ve `<th>` etiketlerinin içerisindeki içeriğe fazladan bir *dolgu alanı* ekler. Bu, tablonun etrafındaki alanı etkili bir şekilde kontrol etmek için kullanılabilir. Bir etiketin dolgusu (padding), *etiketin içeriği ile sınırı arasındaki boşluktur*. Tablo 6.10.'deki `<td>` etiketindeki bildirim “padding:10px”, hücre içeriğinin kenarlarından 10px uzaklıkta olması sağlar. Sayfanın tarayıcıda açıldığında şeklini gösteren bölümde, hücrelerin içeriğinin hücre kenarından 10px mesafede olduğu gözlemlenebilmektedir.

Tablo 6.10. Padding Özelliği Kullanımı (10px)

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü												
<pre><style type="text/css"> table, th, td { border: 1px solid black; } table{ border-collapse:collapse; } td{padding:10px; } </style></pre>	<table border="1"> <thead> <tr> <th>İsim</th> <th>Soyisim</th> <th>Sancar</th> </tr> </thead> <tbody> <tr> <td>Aziz</td> <td>Sancar</td> <td>72</td> </tr> <tr> <td>Mert</td> <td>Gazoz</td> <td>20</td> </tr> <tr> <td>Ali</td> <td>Topaloğlu</td> <td>19</td> </tr> </tbody> </table>	İsim	Soyisim	Sancar	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Sancar											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

Aynı örneğin “`td{padding:1px;}`” bildirimini tarayıcıda gösterimi Şekil 6.4.'teki gibi olacaktır. Şekil 6.4.'te hücrelerin içeriğinin hücre kenarına çok yakın mesafede (1px) olduğu kolayca gözlemlenebilmektedir.

 Bir etiketin dolgusu (padding), etiketin içeriği ile sınırı arasındaki boşluktur.

İsim	Soyisim	Sancar
Aziz	Sancar	72
Mert	Gazoz	20
Ali	Topaloğlu	19

Şekil 6.4. Padding Özelliği Kullanımı (1px)

Padding özelliği Tablo 6.4.'teki `` etiketi örneğinde olduğu gibi çeşitli şekillerde kullanılarak çeşitlendirilebilir. Tablo 6.11.'de `<p>` etiketinin içindeki “Bilgisayar” kelimesine (`` etiketine) “farklılık” class tanımı uygulanmıştır. “farklılık” class tanımında metin color özelliği kırmızı, text-decoration özelliği

için üzeri çizgili, text-transform özelliği için büyük harf, font-style özelliği için eğik, background-color özelliği için "#99DDFF" hexadecimal renk kodu, border özelliği için 1 piksel düz siyah renk ve padding özelliği için 0 piksel değeri bildirilmiştir. Padding özelliğinin değeri 10 piksel olarak bildirilmiş hâlinin tarayıcıda açıldığında görünümü de Şekil 6.5.'te gösterilmektedir. Şekil 6.5.'te ayrıca bütün stil tanımlamaları değiştirmeden sadece padding özelliğinin sıfır eşitlenmiş hâlinin web tarayıcısında görünümü de eklenmiştir. İki web sayfası görüntüsü incelendiğinde padding değeri 10 piksel olarak tanımlanan web sayfası görüntüsündeki içeriğin kendi kenarlarından uzakta yerleştirildiği görülmektedir. Padding değeri sıfır piksel olarak tanımlanan web sayfası görüntüsündeki içeriğin kendi kenarlarına yapışık olduğu görülmektedir. "Farklilik" stil class tanımında, kenarlık kullanarak etiketinin etkilediği alan gösterilmek istenmiştir.

Şekil 6.11. Padding Özelliğinin Farklı Değerlerle Kullanımı

Html Sayfası
<pre><html> <head><title>Stiller</title> <style type="text/css"> .farklilik{color: red; text-decoration: line-through; text-transform: uppercase; font-style: italic; padding:10px; border: 1px solid black; background-color: #99DDFF;} </style> </head> <body> <p>Atatürk Üniversitesi, Açık Öğretim Fakültesi, Bilgisayar Programcılığı Bölümü</p> </body> </html></pre>

web sayfası görüntüsü (padding:10px;)
Atatürk Üniversitesi, Açık Öğretim Fakültesi, BİLGİSAYAR Programcılığı Bölümü
web sayfası görüntüsü (padding:0px;)
Atatürk Üniversitesi, Açık Öğretim Fakültesi, BİLGİSAYAR Programcılığı Bölümü

Şekil 6.5. Padding Özelliğinin Farklı Değerlerle Kullanımı


Padding özelliği, aslında padding-top, padding-right, padding-bottom ve padding-left özellikleri için bir kısa yol özellikleidir.

Padding değer bildiriminde üç değer varsa; örneğin, "padding:15px 10px 5px;" bildiriği yapılmışsa, üst dolgu için 15 piksel, sağ ve sol dolgu için 10 piksel ve alt dolgu için 5 piksel kullanılın demektir. Padding değer bildiriminde iki değer bildiriği de kabul edilmektedir. Örneğin, "padding:10px 5px;" bildiriği yapılmışsa bu üst ve alt dolgu için piksel ve sağ ve sol dolgu için 5 piksel kullanılın demektir.

Padding özelliği, aslında **padding-top**, **padding-right**, **padding-bottom** ve **padding-left** özellikleri için bir kısa yol özellikleidir. Önceki örneklerde **<td>** etiketinin

padding özellik değeri 10 piksel, 1 piksel ve sıfır piksel olarak belirlenmişti, bu bildirim şekliyle her dört taraf dolgu (üst, sağ, alt ve sol) için aynı değer bildirilmişti. Dolgu sıralaması üst değerinden bağlayıp saat yönünde ilerlemektedir. Önceki örnekte kullanılan padding özelliği bildirimi olan “td{padding:1px;}”, “td{padding:1px 1px 1px 1px;}” şeklinde de yapılabildi. Padding stil bildirimi her yön için farklı olarak; örneğin “td{padding:15px 10px 5px 0px;}” bildirilmek istenirse, bu bildirim ile üst dolgu için 15 piksel, sağ dolgu için 10 piksel, alt dolgu için 5 piksel ve sol dolgu için 0 piksel değeri kullanılın demektir. Tablo 6.12.'de bildirim şekli ve bildirim sonucunun tarayıcıda gösterimi yapılmıştır. <td> etiketi kullanılan bölgelerde (hücrelerde) içeriğin üst çizgiden uzaklığının diğer kenarlardan daha çok olduğu ve sol kenarda ise içeriğin kenara yaslı (0px) olduğu görülmektedir.

Tablo 6.12. Padding Özelliği Kullanımı Şekli

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü		
<pre><style type="text/css"> table,th, td { border: 1px solid black;} table{ border-collapse: collapse;} td {padding:15px 10px 5px 0px;} </style></pre>	İsim	Soyisim	Yaş
	Aziz	Sancar	72
	Mert	Gazoz	20
	Ali	Topaloğlu	19

Padding değerleri bütün yönler için bir defada bütün olarak verilmek zorunda değildir. *Her bir yön için ayrı ayrı stil bildirimi yapılabılır*. En son kullanılan bildirim, her yön için ayrı bildirim tanımlanarak da yapılabildi (td {padding-top: 15px; padding-right: 10px ; padding-bottom: 5px; padding-left: 0px;}). Tablo 6.13.'te bildirim yöntemi ve bildirim sonucunun tarayıcıda gösterimi yapılmıştır. Üstteki Tablo 6.13.'te web sayfası görüntüsünde olduğu gibi burada da bölgelerde (hücrelerde) içeriğin üst çizgiden uzaklığının diğer kenarlardan daha çok olduğu ve sol kenarda ise içeriğin kenara yaslı (0px) olduğu görülmektedir.



Her bir yön için ayrı ayrı padding stil bildirimi yapılabılır.

Tablo 6.13. Padding Özelliğinin Her Kenar İçin Ayrı Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü		
<pre><style type="text/css"> table,th, td { border: 1px solid black;} table{ border-collapse: collapse;} td {padding-top: 15px; padding-right: 10px; padding-bottom: 5px; padding-left: 0px;} </style></pre>	İsim	Soyisim	Yaş
	Aziz	Sancar	72
	Mert	Gazoz	20
	Ali	Topaloğlu	19

Stil tanımında bütün yönler için padding bildirimi yapılmış olmazsa, *bildirim yapılmayan yönler için varsayılan değerler* kullanılır. Tablo 6.14.'te üst padding bildirimi 10 pixel (td {padding-top: 15px;}) ve alt padding bildirimi 20 pixel (td

{padding-bottom: 20px;}) tanımlanması ve bu işlemin sonucunun tarayıcıda gösterimi yapılmıştır. İşlem sonucunda üst ve alt dolgu için bildirimde kullanılan değerler, sol ve sağ dolgu için tarayıcının varsayılan değerleri kullanılmıştır. "Topaloğlu" yazan hücrenin sağ ve sol tarafında da görüldüğü gibi varsayılan değerlerin kullanıldığı daha açık bir şekilde görülmektedir.

Tablo 6.14. Padding Özelliğinin Tek Kenar İçin Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü		
	İsim	Soyisim	Yaş
<style type="text/css"> table, th, td { border: 1px solid black; } table{ border-collapse: collapse; } td {padding-top: 15px; } td {padding-bottom: 20px; } </style>	Aziz	Sancar	72
	Mert	Gazoz	20
	Ali	Topaloğlu	19



Stil tanımında bütün yönler için padding bildirimi yapılmamışsa, bildirim yapılmayan yönler için varsayılan değerler kullanılır.

Kenar Boşlukları (margin)

CSS margin özelliği, *etiketlerin etrafında, tanımlanmış sınırların dışında boşluk oluşturmak için kullanılır*. Margin özelliği ile kenar boşlukları üzerinde tam kontrol kullanıcıda olmaktadır. Bir etiketin her bir tarafı için kenar boşluğunu ayarlama özellikleri (üst, sağ, alt ve sol) bulunmaktadır. Margin özelliğinde, padding özelliğinde olduğu gibi tek bildirimle bütün yönlerdeki kenar boşluk miktarını belirlemek (margin:10px;), her bir yönde kenar boşluk miktarını ayrı ayrı belirlemek (p {margin:20px 10px 5px 0px;}) veya sadece tek bir yönde kenar boşluk miktarını belirlemek (margin-top:10px;) mümkündür. Margin özellikleri aşağıdaki değerlere sahip olabilmektedir:

- cm, px, pt, vb. uzunluk değerleri,
- yüzdelik değerleri ve
- tarayıcı tarafından hesaplanan otomatik değerler.

Tablo 6.15.'teki örnekte bir <p> etiketi için border (kenarlık) ve margin (kenar boşluğu) özellikleri için stil bildirimi yapılmıştır. <p> etiketinin border özelliği için bir piksel kalınlığında, düz ve kırmızı değer bildirimi yapılmıştır. <p> etiketinin margin özelliği için ise 10 piksel değer bildirimi yapılmıştır. Tablo 6.15.'te ayrıca <p> etiketinin bir hücredeki içeriğe uygulanması da gösterilmiştir. Şekil 6.17.'de, dosyanın tarayıcıda açıldığı hâli de gösterilmektedir. Bu görüntüde <p> etiketinin içeriğinin (sınırları kırmızı çizgi ile gösterilmiştir) içinde bulunduğu tablo hüresinin kenarlarından eşit uzaklıkta olduğu görülmektedir.

Tablo 6.15. Margin Özelliği Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü	

<pre> <style type="text/css"> table, th, td { border: 1px solid black; } p{border: 1px solid red; margin: 10px;} </style> ... <td> <p>Ali</p></td> ... </pre>	<table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>İsim</th><th>Soyisim</th><th>Yaş</th></tr> </thead> <tbody> <tr> <td>Aziz</td><td>Sancar</td><td>72</td></tr> <tr> <td>Mert</td><td>Gazoz</td><td>20</td></tr> <tr> <td style="background-color: red;">Ali</td><td>Topaloğlu</td><td>19</td></tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

Tablo 6.16.'da aynı içeriğin `<p>` etiketinin *margin özelliği için farklı kenar uzunlukları değerleri bildirimi* yapılması ve bu işlemin tarayıcıda sonucu gösterilmektedir. Bildirimde üst kenarlık için 20 piksel, sağ kenarlık için 10 piksel, alt kenarlık için 5 piksel ve sol kenarlık için 0 piksel değeri belirlenmiştir. Bildirimin tarayıcıda görüntülenmesinde `<p>` etiketinin (daha iyi gözlemlenebilmesi için `<p>` etiketine kenarlık eklenmiştir) hücreden üstten 20 piksel, sağdan 10 piksel, alttan 5 piksel ve soldan 0 piksel mesafede olduğu (piksel değerlerini çıplak gözle tam olarak fark etmek mümkün değildir, ama değerlerin farklı olduğu kolayca anlaşılmaktadır) görülebilmektedir.

```

<style type="text/css">
    table, th, td { border: 1px solid black; }
    p{border: 1px solid red; margin: 10px;}
</style>

```

Tablo 6.16. Margin Özelliğinin Köşeler İçin Farklı Değerlerle Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü												
<pre> <style type="text/css"> table, th, td { border: 1px solid black; } p{border: 1px solid red; margin: 20px 10px 5px 0px;} </style> ... <td> <p>Ali</p></td> ... </pre>	<table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>İsim</th> <th>Soyisim</th> <th>Yaş</th> </tr> </thead> <tbody> <tr> <td>Aziz</td> <td>Sancar</td> <td>72</td> </tr> <tr> <td>Mert</td> <td>Gazoz</td> <td>20</td> </tr> <tr> <td style="background-color: red;">Ali</td> <td>Topaloğlu</td> <td>19</td> </tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

Margin özelliği için stil tanımlamada *farklı sayıda değerler* de kullanılabilmektedir. Değer bildiriminde üç değer varsa örneğin, “margin: 15px 10px 5px;” bildirimini yapılmışsa, bu ifade ile üst dolgu için 15 piksel, sağ ve sol dolgu için 10 piksel ve alt dolgu için 5 piksel kullanılın demektir. Margin değer bildiriminde iki değer bildirimini de kabul edilmektedir. Örneğin, “padding: 10px 5px;” bildirimini yapılmışsa, bu üst ve alt dolgu için 10 piksel ve sağ ve sol dolgu için 5 piksel kullanılın demektir.



Margin özelliği için stil tanımlamada farklı sayıda değerler de kullanılabilir.

Margin özelliği, aslında *margin-top*, *margin-right*, *margin-bottom* ve *margin-left* özellikleri için bir kısa yol özelliğidir. Önceki örneklerdeki "margin: 10px;" stil bildirim şekliyle her dört taraf kenar boşluğu (üst, sağ, alt ve sol) için aynı değer kullanılsın demektir. Diğer örnekteki "margin:20px 15px 10px 0px;" stil bildirim şekliyle tek seferde bütün kenar boşlukları için bildirim yapılmıştır. Margin değerleri bütün yönler için bir defada bütün olarak verilmek zorunda değildir. Her bir yön için ayrı ayrı stil bildirimini de yapılabilmektedir. En son kullanılan her yön için farklı kenar boşluğu bildirimini, her yön için ayrı bildirim tanımlanarak da yapılabiliirdi (`p {margin-top: 15px; margin-right: 10px; margin-bottom: 5px; margin-left: 0px;}`). Tablo 6.17'de her bir yönde kenar boşluğu için özel bildirim tanımlanması ve bildirim sonucunun tarayıcıda gösterimi yapılmıştır. Margin değerinin daha iyi gözlenebilmesi için `<p>` etiketinin kenarlarına çizgi de eklenmiştir. Üstteki Tablo 6.16.'daki web sayfası görüntüsünde olduğu gibi burada da `<p>` etiketinin üst kenardan uzaklığının diğer kenarlardan uzaklığından daha çok olduğu ve sol kenarda ise içeriğin kenara yaslı (0px) olduğu görülmektedir.

Tablo 6.17. Margin Özelliğinin Her Köşe İçin Ayrı Ayrı Tanımlanması

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü												
<pre><style type="text/css"> table, th, td, p { border: 1px solid black;} p {margin-top: 15px; Margin-right: 10px; margin-bottom: 5px; margin-left: 0px;} </style> ... <td> <p>Ali</p></td> ...</pre>	<table border="1"> <thead> <tr> <th>İsim</th> <th>Soyisim</th> <th>Yaş</th> </tr> </thead> <tbody> <tr> <td>Aziz</td> <td>Sancar</td> <td>72</td> </tr> <tr> <td>Mert</td> <td>Gazoz</td> <td>20</td> </tr> <tr> <td>Ali</td> <td>Topaloğlu</td> <td>19</td> </tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

Stil tanımında bütün yönler için margin bildirimini yapılmış olmazsa, *bildirim yapılmayan yönler için varsayılan değerler* kullanılır. Tablo 6.18.'de üst margin özelliği bildirimini 10 pixel (`p {margin-top:10px;}`) ve alt margin özelliği bildirimini 20 pixel (`p {margin-bottom:20px;}`) tanımlanması ve bu işlemin sonucunun tarayıcıda gösterimi yapılmıştır. İşlem sonucunda ("Ali" yazan hücrenin) üst ve alt dolgu için bildirimde tanımlanan değerler, sol ve sağ dolgu için tarayıcının varsayılan değerleri kullanılmıştır. Hücrenin sağ ve sol tarafında da görüldüğü gibi varsayılan değerlerin kullanıldığı daha açık bir şekilde görülmektedir.

Tablo 6.18. Margin Özelliğinin Tek Kenar İçin Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
--------------	------------------------------------

```

<style type="text/css">
    table, th, td, p { border: 1px solid black; }
    p { margin-top: 10px; margin-bottom: 20px; }
</style>
...
<td> <p>Ali</p></td>
...

```

İsim	Soyisim	Yaş
Aziz	Sancar	72
Mert	Gazoz	20
Ali	Topaloğlu	19

Margin ve Padding

Padding (dolgu), bir elemanın içinde fazladan boşluk yaratırken, margin (kenar boşluğu) bir elemanın etrafında fazladan boşluk yaratır. Bir örnekte margin ve padding değeri birlikte kullanılırsa ayırmayı yapmak daha kolaylaşabilecektir. Tablo 6.19.'da <p> etiketi CSS tanımına padding-left özelliğinin değeri 25 piksel bildirim yapılmıştır. Margin özelliği için de üstten 20 piksel, sağdan 15 piksel, alttan 10 piksel ve sağdan 0 piksel değer bildirimleri yapılmıştır. Tablo 6.19.'da ayrıca işlem sonucunun tarayıcıda görüntülenmesi de bulunmaktadır. Görüntü incelediğinde <p> etiketi içeriğinin hücrenin üst, sağ, alt ve sol köşelerinden farklı mesafelere yerleştirildiği görülmektedir. Bu düzenlemeyi elde etmek için margin (kenar boşluğu) özelliği kullanılmıştır. <p> etiketinin içeriği incelediğinde etiket içeriğinin etiketin sol köşesinden 25 piksel içерden başladığı görülmektedir. Bir başka ifade ile içeriğin <p> alanı için belirlenen alandan sol kenarından 25 piksel içерden (dolgu kullanılmıştır) başlamıştır. Bu görüntüyü elde etmek için ise padding özelliği kullanılmıştır.

```

<style type="text/css">
    table, th, td { border: 1px solid black; }
    p { border: 1px solid red; margin: 20px 10px 5px 0px;
        padding-left: 25px; }
</style>

```

Tablo 6.19. Margin ve Padding Özelliğinin Birlikte Kullanımı

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü												
<style type="text/css"> table, th, td { border: 1px solid black; } p { border: 1px solid red; margin: 20px 10px 5px 0px; padding-left: 25px; } </style> ... <td> <p>Ali</p></td> ...	<table border="1"> <thead> <tr> <th>İsim</th><th>Soyisim</th><th>Yaş</th></tr> </thead> <tbody> <tr> <td>Aziz</td><td>Sancar</td><td>72</td></tr> <tr> <td>Mert</td><td>Gazoz</td><td>20</td></tr> <tr> <td>Ali</td><td>Topaloğlu</td><td>19</td></tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

**Bireysel Etkinlik**

- Padding ve margin özelliklerini gösterebilecek bir örnek hazırlayınız.
- Örnek üzerinde padding ve margin özellikleri kolayca anlaşılmalıdır.

Tablo Rengi

Tabloyu görsel olarak daha çekici hâle getirmek için *tablodaki arka plan rengi ve metnin rengi* değiştirebilir. Arka plan rengi için background-color özelliği ve metin rengi için color özelliği kullanılır. Tablo 6.20.'de <td> ve <th> etiketleri için arka plan ve metin rengi bildirimlerine örnek verilmiş ve bildirim sonucu tarayıcıda gösterilmiştir. <th> etiketi için stil tanımlamasında arka plan rengi gri ve metin rengi beyaz olarak bildirilmiştir. <td> etiketi için stil tanımlamasında arka plan rengi olarak rgb(0, 10, 255) gri ve metin rengi beyaz olarak bildirilmiştir.

Tablo 6.20. Tablo Rengi İçin İçerik Tanımlama

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü												
<pre><style type="text/css"> table, th, td { border: 1px solid black;} table {border-collapse: collapse;} th{background-color: grey; color: white;} td {background-color: rgb(0, 10, 225); color: white;} </style></pre>	<table border="1"> <thead> <tr> <th>İsim</th> <th>Soyisim</th> <th>Yaş</th> </tr> </thead> <tbody> <tr> <td>Aziz</td> <td>Sancar</td> <td>72</td> </tr> <tr> <td>Mert</td> <td>Gazoz</td> <td>20</td> </tr> <tr> <td>Ali</td> <td>Topaloğlu</td> <td>19</td> </tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

Hover Seçici

Hover seçici, fare üzerine getirildiğinde *belirli bir hücreyi veya sırayı vurgulamayı* sağlar. Örnek olarak imleç tabloda satırların üzerine geldiğinde satırındaki hücrelerin arka plan rengini sarı yapan stil bildirim “tr : hover { background-color : yellow ;}” ve imleç tabloda hücrelerin üzerine geldiğinde hücrelerdeki metinleri büyük harf gösteren bildirim “td : hover { text-transform : uppercase; }” olarak tanımlanabilir. Tablo 6.21.'de <th> ve <td> etiketleri için hover seçici css bildirimleri ve sonuçlarının tarayıcıda gösterimi yapılmıştır. Tablo 6.21.'deki web sayfası görüntüsü incelendiğinde, imlecin üzerinde olduğu satırının arka plan renginin sarı ve bu satırındaki imlecin üzerinde olduğu hücrenin içeriğinin büyük harflerle gösterildiği gösterilmektedir.



Hover seçici, fare üzerine getirildiğinde belirli bir hücreyi veya sırayı vurgulamayı sağlar.

Tablo 6.21. Tablo İçin Hover Özelliği Kullanımı

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü												
<pre><style type="text/css"> table, th, td { border: 1px solid black;} table{ border-collapse:collapse;} tr:hover{background-color:yellow;} td:hover{text-transform:uppercase;} </style></pre>	<table border="1"> <thead> <tr> <th>İsim</th><th>Soyisim</th><th>Yaş</th></tr> </thead> <tbody> <tr> <td>Aziz</td><td>Sancar</td><td>72</td></tr> <tr> <td>Mert</td><td>Gazoz</td><td>20</td></tr> <tr style="background-color: yellow;"> <td>Ali</td><td>TOPALOĞLU</td><td>19</td></tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	TOPALOĞLU	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	TOPALOĞLU	19											

Şeritli Tablo

Şeritli tablo oluşturmak için “*nth-child*” seçicisi kullanılabilir. Tek veya çift numaralı tablo sırası seçilerek ikisinden birinin veya ikisinin de arka plan rengi değiştirebilir. Büyük tablolar olduğunda, şeritli tablo kullanmak iyi bir uygulamadır, *izleyicinin aradığı içeriği kolayca aramasında ve satırları gözüyle zorlanmadan takip etmesinde* yardımcı olur. Değer olarak sayı (tablonun kaçinci satırı olduğunu belirtmek için), *odd* (bütün tek numaralı satırları belirtmek için), *even* (bütün çift numaralı satırları belirtmek için) ve *n* (tam sayıları- çeşitli şekillerde formüller oluşturmak için) kullanılabilir. Tablo 6.22.’deki css bildiriminde tablonun ikinci satırını belirtmek için “tr : nth-child (2);” ve satırın arka plan rengini belirtmek için “{background-color : grey ;}” ifadesi kullanılmıştır. Tablo 6.22.’de dosyanın tarayıcıda açıldığında bildirimin etkisi de görülmektedir. Şekil incelendiğinde ikinci satırın arka plan renginin bildirimde tanımlandığı gibi gri olduğu görülmektedir.

Tablo 6.22. Şeritli Tablo Kullanımı

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü												
<pre><style type="text/css"> table th, td{border: 1px solid black;} table{border-collapse:collapse;} tr:nth-child(2){background-color:grey;} </style></pre>	<table border="1"> <thead> <tr> <th>İsim</th><th>Soyisim</th><th>Yaş</th></tr> </thead> <tbody> <tr style="background-color: grey;"> <td>Aziz</td><td>Sancar</td><td>72</td></tr> <tr> <td>Mert</td><td>Gazoz</td><td>20</td></tr> <tr> <td>Ali</td><td>Topaloğlu</td><td>19</td></tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloğlu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloğlu	19											

 Tablodaki tek numaralı satırları (tablodaki satır numaraları bir ile başlamaktadır) belirtmek için *odd* değeri kullanılmaktadır. Tablo 6.23.’teki bildirimde odd değerinin kullanımı ve dosyanın tarayıcıda açıldığında bildirimin etkisi görülmektedir. Stil tanımında tek numaralı (*odd*) satırların arka plan renginin gri olması bildirilmiştir. Tablo 6.23. incelendiğinde birinci satırın (başlık satırı) ve üçüncü satırın arka plan renginin bildirimde tanımlandığı gibi gri olduğu görülmektedir.

Tablo 6.23. Şeritli Tablo Kullanımı (Tek /Çift)

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü												
<pre><style type="text/css"> table th, td{border: 1px solid black;} table{border-collapse:collapse;} tr:nth-child(odd){background-color:grey;} </style></pre>	<table border="1"> <thead> <tr> <th>İsim</th> <th>Soyisim</th> <th>Yaş</th> </tr> </thead> <tbody> <tr> <td>Aziz</td> <td>Sancar</td> <td>72</td> </tr> <tr> <td>Mert</td> <td>Gazoz</td> <td>20</td> </tr> <tr> <td>Ali</td> <td>Topaloglu</td> <td>19</td> </tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloglu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloglu	19											

nth-child seçici bildiriminde “*n*” karakteri tam sayıları (0, 1, 2, 3, ...) belirtmek için kullanılabilir. Tablo 6.24.'teki bildirimde değer olarak “2n+2” kullanılmıştır. Bu ifade ile tarayıcıya tabloda satır olduğu sürece “n” karakteri yerine tam sayıları kullanarak işlem yap denilmektir. İlk olarak “n” karakteri yerine 0 konursa “2*0+2” işleminin değeri “2” olur. Tabloda 2 numaralı satır vardır ve arka plan rengi gri yapılacaktır. Sonraki adımda “n” karakteri yerine 1 konursa “2*1+2” işleminin değeri “4” olur. Tabloda 6.24.'te dört numaralı satır son satırıdır, bu satırın arka plan rengi de gri yapılacaktır. Dosya tarayıcıda açıldığında bildirimin etkisi görülmektedir. Tablo 6.24. incelendiğinde ikinci satırın ve dördüncü satırın arka plan renginin bildirimde tanımlandığı gibi gri olduğu görülmektedir. Tabloda daha fazla satır olsayıdı “n” karakteri yerine sırasıyla tam sayılar konularak benzer şekilde işlem yapılmaya devam edilecekti.

Tablo 6.24. Şeritli Tablo Kullanımı (n'inci Sayı)

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü												
<pre><style type="text/css"> table th, td{border: 1px solid black;} table{border-collapse:collapse;} tr:nth-child(2n+2){background- color:grey;} </style></pre>	<table border="1"> <thead> <tr> <th>İsim</th> <th>Soyisim</th> <th>Yaş</th> </tr> </thead> <tbody> <tr> <td>Aziz</td> <td>Sancar</td> <td>72</td> </tr> <tr> <td>Mert</td> <td>Gazoz</td> <td>20</td> </tr> <tr> <td>Ali</td> <td>Topaloglu</td> <td>19</td> </tr> </tbody> </table>	İsim	Soyisim	Yaş	Aziz	Sancar	72	Mert	Gazoz	20	Ali	Topaloglu	19
İsim	Soyisim	Yaş											
Aziz	Sancar	72											
Mert	Gazoz	20											
Ali	Topaloglu	19											



Özet

• RENK

• Renkler, ilgili herhangi bir CSS kodu kullanılarak web sitesine veya blog'a uygulanabilirler. Metin rengini ayarlamak için color, arka plan rengini ayarlamak için background-color ve kenarlık rengini ayarlamak için border-color özelliği kullanılmaktadır.

• **Renk Adı:** Renk adı doğrudan beyan edilerek renk bildirilebilir. Adlandırılmış renkler İngilizce isimleriyle bildirilmek zorundadırlar ve büyük / küçük harfe duyarlı degillerdir.

• **Onaltılık Kod (Hexadecimal Code):** Renk, her 2 hanenin kırmızı, yeşil, mavi renklerini temsil ettiği 6 basamaklı onaltılık sayı kullanılarak bildirilebilir.

• **RGB Kodu:** İstenilen rengi elde etmek için ne kadar kırmızı, yeşil ve mavi rengin kullanıldığı bildirilen RGB yöntemi kullanılarak da renk seçimi yapılabilir.

• **Renk (color):** Renk özelliği, metnin ön plan rengini değiştirmek için kullanılır. Color özelliği bütün tarayıcıların ilk sürümünden itibaren desteklenmektedir.

• **Arka Plan (background):** Arka plan renklerini bildirmek için background-color özelliği kullanılır.

• SPAN

• etiketi, satır içi öğeler ve içerik için genel bir satır içi kapsayıcıdır.

• TABLO DÜZENLEME

• **Kenar (border):** CSS'de tablo kenarlığını belirtmek için border özelliği kullanılır. Border özelliğinin, border-width (kenarlık genişliği), border-style (kenarlık stili) ve border-color (kenarlık rengi) olmak üzere üç alt özelliği bulunmaktadır.

• **Kenar Daraltma (border-collapse):** Tablo kenarlıklarının tek bir kenarlığa daraltılması veya standart HTML'deki gibi mi ayrılacağını belirler. Tablo hücreleri etrafındaki kenarlık ile olan mesafeyi düzenlemeye olanak sağlar.

• **Yükseklik ve Genişlik (height ve width):** Tablonun ve tablodaki hücrelerin genişlik ve yüksekliği, height ve width özellikleri tarafından tanımlanır. Yüksekliği ve genişliği bildirmek için yüzdelik değerler (%), em) veya tam değerler (piksəl, pt, cm) kullanılabilir.

• **Dolgu (padding):** Padding özelliği, <td> ve <th> etiketlerinin içerisindeki içeriğe fazladan bir dolgu alanı ekler. Bir etiketin dolgusu (padding), etiketin içeriği ile sınırı arasındaki boşluktur. Padding özelliği, aslında padding-top, padding-right, padding-bottom ve padding-left özellikleri için bir kisa yol özelliğidir.

• **Kenar Boşlukları (margin):** CSS margin özelliği, etiketlerin etrafında, tanımlanmış sınırların dışında boşluk oluşturmak için kullanılır. Margin özelliği ile kenar boşlukları üzerinde tam kontrol kullanıcıda olmaktadır. Bir etiketin her bir tarafı için kenar boşluğunu ayarlama özellikleri bulunmaktadır.

• **Margin ve Padding:** Padding (dolgu), bir eleman içinde fazladan boşluk yaratırken, margin (kenar boşluğu) bir elemanın etrafında fazladan boşluk yaratır.

• **Tablo Rengi:** Tabloyu görsel olarak daha çekici hale getirmek için tablodaki arka plan rengi ve metnin rengi değiştirebilir. Arka plan rengi için background-color özelliği ve metin rengi için color özelliği kullanılır.

• **Hover Seçici:** Hover seçici, fare üzerine getirildiğinde belirli bir hücreyi veya sırayı vurgulamayı sağlar.

• **Şeritli Tablo:** Şeritli tablo oluşturmak için "nth-child" seçicisi kullanılabilir. Tek veya çift numaralı tablo sırası seçilerek ikisinden birinin veya ikisinin de arka plan rengi değiştirebilir.

DEĞERLENDİRME SORULARI

1. Aşağıda yer alan CSS'te renk ile ilgili ifadelerden hangisi doğru değildir?
 - a) Adlandırılmış renkler İngilizce isimleriyle ifade edilirler.
 - b) HTML sayfalarında kullanılmak üzere 140 tane adlandırılmış renk vardır?
 - c) Bütün adlandırılmış renkler aynı zamanda hexadecimall kod olarak da bildirilebilirler.
 - d) RGB renk kodu bildiriminde opaklık değeri kullanmak zorunlu değildir.
 - e) Hexadecimal renk kodu tanımında her bir rengi temsil eden değerler 0 ile 255 arasındadır.
2. Aşağıdaki kod web tarayıcısında açılırsa nasıl görünür?

```
<p style="text-decoration:underline; font-style:italic;">  
Atatürk Üniversitesi  
<span style="font-style:normal; background:yellow;">Açık </span>  
Öğretim fakültesi </p>
```

Atatürk Üniversitesi Açık Öğretim fakültesi

a)

Atatürk Üniversitesi Açık Öğretim fakültesi

b)

Atatürk Üniversitesi Açık Öğretim fakültesi

c)

Atatürk Üniversitesi Açık Öğretim fakültesi

d)

Atatürk Üniversitesi Açık Öğretim fakültesi

e)

3. Aşağıdaki görüntüyü elde etmek için hangi kod bloku kullanılmış olabilir?

Atatürk Üniversitesi

Açık Öğretim fakültesi

- a)

```
<p style="border:1px solid black;"> Atatürk Üniversitesi </p>  
<p style="border:1px solid black; padding-left:30px; padding-top:10px">Açık Öğretim fakültesi </p>
```
- b)

```
<p style="border:1px solid black;"> Atatürk Üniversitesi </p>  
<p style="border:1px solid black; width:30px; height:10px">Açık Öğretim fakültesi </p>
```
- c)

```
<p style="border:1px solid black; width:30px;"> Atatürk Üniversitesi </p>  
<p style="border:1px solid black; height:10px">Açık Öğretim fakültesi </p>
```
- d)

```
<p style="border:1px solid black; height:10px;"> Atatürk Üniversitesi </p>  
<p style="border:1px solid black; width:30px;">Açık Öğretim fakültesi </p>
```
- e)

```
<p style="border:1px solid black;"> Atatürk Üniversitesi </p>  
<p style="border:1px solid black; margin-left:30px; margin-top:10px">Açık Öğretim fakültesi </p>
```

4. Aşağıdaki renk tanımlarından hangisi geçersizdir?
- a) `<p style="color:#FF0000;"> Atatürk Üniversitesi </p>`
 - b) `<p style="color:hsl(120, 250, 150, 0.5);> Atatürk Üniversitesi </p>`
 - c) `<p style="color:red;"> Atatürk Üniversitesi </p>`
 - d) `<p style="color:RGB(255,0,0, 0.5);> Atatürk Üniversitesi </p>`
 - e) `<p style="color:RGB(255,0,0);> Atatürk Üniversitesi </p>`

5. Aşağıdaki görüntüyü elde etmek için hangi kod bloku kullanılabilir?

Atatürk Üniversitesi

- a) `<p style="border:1px solid red; width:200px; margin:30px; padding-left: 30px;"> Atatürk Üniversitesi </p>;`
- b) `<p style="border:1px solid red; width:200px; margin-bottom:30px; padding-left: 30px;"> Atatürk Üniversitesi </p>;`
- c) `<p style="border:1px solid red; width:200px; margin-bottom:30px; padding: 30px;"> Atatürk Üniversitesi </p>;`
- d) `<p style="border:1px solid red; width:200px; margin-left:30px; padding-top: 30px;"> Atatürk Üniversitesi </p>;`
- e) `<p style="border:1px solid red; height:200px; margin-bottom:30px; padding-left: 30px;"> Atatürk Üniversitesi </p>;`

6. Aşağıdaki tablo görüntüsünü elde etmek için hangi css tanımı kullanılmış olabilir?

İsim	Soyisim	Yaş
Aziz	Sancar	72
Mert	Gazoz	20
Ali	Topaloğlu	19

- a)

```
<style type="text/css"> table{border-bottom: 1px solid red;  
border-top:1px solid red;}</style>
```
- b)

```
<style type="text/css"> th{border-bottom: 1px solid red;  
border-top:1px solid red;}</style>
```
- c)

```
<style type="text/css"> td{border-bottom: 1px solid red;  
border-top:1px solid red;}</style>
```
- d)

```
<style type="text/css"> tr{border-bottom: 1px solid red;  
border-top:1px solid red;}</style>
```
- e)

```
<style type="text/css"> table{border: 1px solid red;  
border:1px solid red;}</style>
```

7. Aşağıdaki içeriği elde etmek için tablo kullanılmıştır. Şekildeki gibi ilk satırdan itibaren, tek numaralı satırların arka plan rengini değiştirmek için hangi css tanımı kullanılmış olabilir? Css tanımı daha fazla satır olsaydı, tek numaralı satırların arka plan rengini de otomatik olarak değiştirmelidir.

İsim	Soyisim	Yaş
Aziz	Sancar	72
Mert	Gazoz	20
Ali	Topaloğlu	19

- a)

```
<style type="text/css"> td:nth-child(odd) {background-color:#FFAA00 ;}
```
- b)

```
<style type="text/css"> table:nth-child(odd) {background-color:#FFAA00 ;}<sty
```
- c)

```
<style type="text/css"> tr:nth-child(odd) {background-color:#FFAA00 ;}<style>
```
- d)

```
<style type="text/css"> tr:nth-child(even) {background-color:#FFAA00 ;}
```
- e)

```
<style type="text/css"> tr:nth-child(even) {background-color:#FFA
```

8. Aşağıdaki içeriği elde etmek için tablo kullanılmıştır. Şekildeki gibi imleç üzerinde geldiğinde hücrenin arka plan rengini gümüş (silver), kenar rengini turuncu (orange) ve 3 piksel yapmak için hangi css tanımı kullanılmış olabilir?

İsim	Soyisim	Yaş
Aziz	Sancar	72
Mert	Gazoz	20
Ali	Topaloğlu	19

- a) `<style type="text/css"> td:hover{background-color:silver; border:3px inset orange;}</style>`
- b) `<style type="text/css"> td:hover{color:silver; border:3px inset orange;}</style>`
- c) `<style type="text/css"> td:hover{background-color:orange; border:3px inset silver;}</style>`
- d) `<style type="text/css"> tr:hover{background-color:orange; border:3px inset silver;}</style>`
- e) `<style type="text/css"> tr:hover{color:silver; border:3px inset orange;}</style>`

9. Aşağıdaki içeriği elde etmek için tablo kullanılmıştır. Tablodaki herhangi bir hücrenin imleç üzerine geldiğinde, hücrenin şekildeki gibi görünmesini sağlamak için hangi css tanımı kullanılmış olabilir?

İsim	Soyisim	Yaş
Aziz	Sancar	72
Mert	Gazoz	20
Ali	Topaloğlu	19

- a) `<style type="text/css"> td:hover{background-color:silver; margin:30px; border:3px inset orange;}</style>`
- b) `<style type="text/css"> td:hover{background-color:silver; padding:30px; border:3px inset orange;}</style>`
- c) `<style type="text/css"> td:hover{background-color:silver; height:30px; border:3px inset orange;}</style>`
- d) `<style type="text/css"> td:hover{background-color:silver; width:30px; border:3px inset orange;}</style>`
- e) `<style type="text/css"> td:hover{background-color:silver; font-weight:800; border:3px inset orange;}</style>`

10. Aşağıdaki görüntüyü elde etmek için hangi kod bloku kullanılmış olabilir?

AOF

- a) <p style="height:50px; width:150px; border:1px dashed orange; margin:20px 0px 0px 0px; font-weight:bold;">AOF</p>
- b) <p style="width:50px; height:150px; border:1px dashed orange; padding:20px 0px 0px 0px; font-weight:bold;">AOF</p>
- c) <p style="height:50px; width:150px; border:1px solid orange; padding:20px 0px 0px 0px; font-weight:bold;">AOF</p>
- d) <p style="height:50px; width:150px; border:1px dashed orange; padding:20px 0px 0px 0px; font-weight:bold;">AOF</p>
- e) <p style="height:50px; width:150px; border:1px dashed orange; padding:0px 0px 0px 20px; font-weight:bold;">AOF</p>

Cevap Anahtarı

1.e, 2.d, 3.e, 4.b, 5.b, 6.a, 7.c, 8.a, 9.c, 10.d

YARARLANILAN KAYNAKLAR

- Brown, T. B. (2018). CSS master (2nd Edition). VIC: SitePoint Pty. Ltd.
- Dean, J. (2019). Web programming with HTML5, CSS, and Javascript. MA: Jones & Bartlett Learning, LLC, an Ascend Learning Company.
- Grant, K. J. (2018). CSS in Depth. NY: Manning Publications Co.
- Meyer, E. A. (2018). CSS Pocket Reference (5th Edition). CA: O'Reilly Media, Inc.

LİSTE VE GÖRSELLERDE CSS STİLLERİ

İÇİNDEKİLER

- Listeler
- Görseller
 - images
 - border
 - border-image
 - border-radius
 - opacity
 - boyutlar
 - background-image



HEDEFLER

- Bu üniteyi çalıştından sonra;
- Listelerle ilgili stillerin neler olduğunu söyleyebilecek,
- Listelerle ilgili css bildirimlerinin sonuçlarını seçebilecek,
- Listelerle ilgili görüntüleri oluşturabilecek ve css tanımlarını gösterebilecek,
- Görsellerle ilgili özellikleri açıklayabilecek,
- Görsellerle ilgili css bildirimlerinin sonuçlarını seçebilecek,
- Görsellerle ilgili görüntüleri oluşturabilecek ve css tanımlarını gösterebileceksiniz.



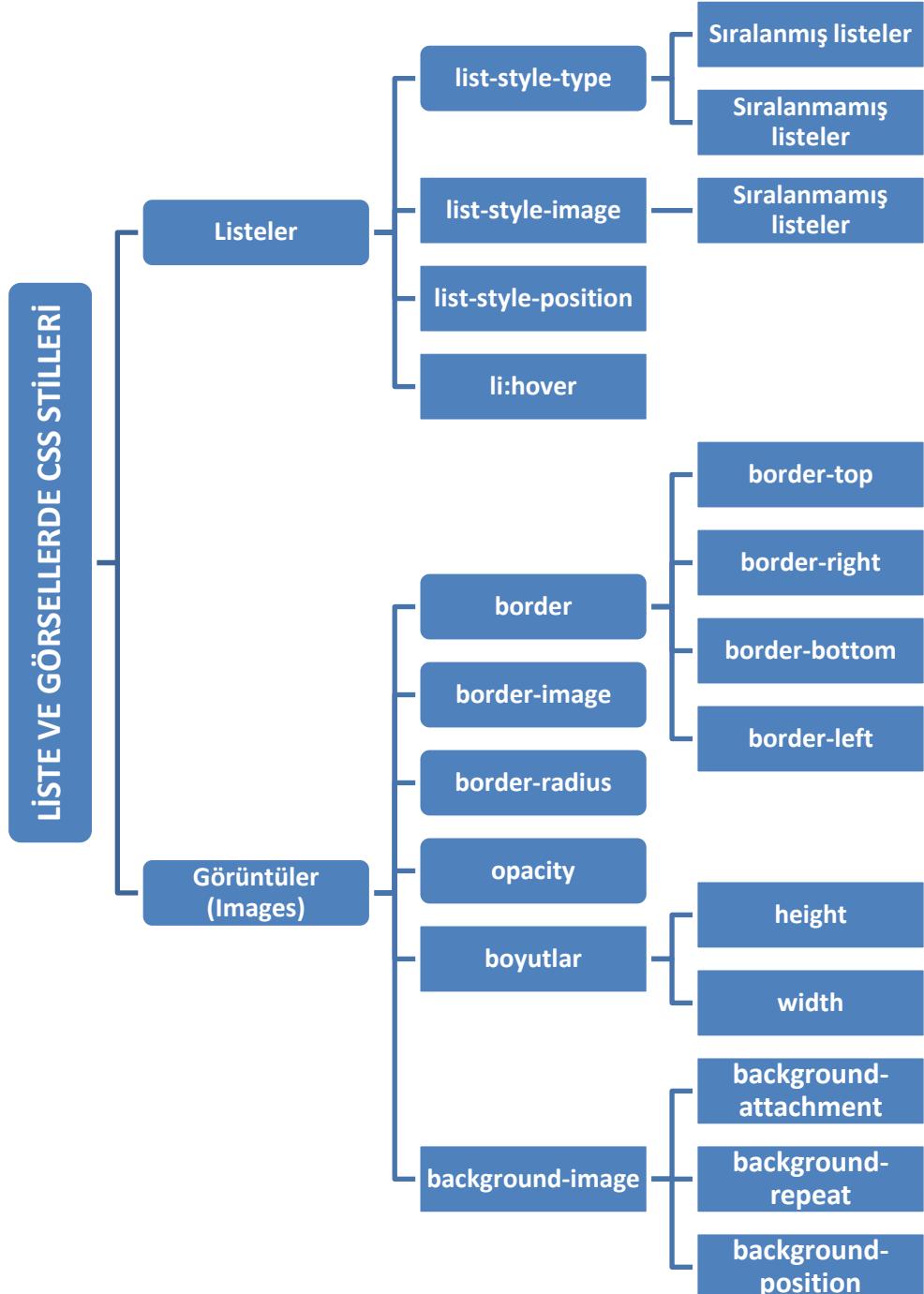
Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET PROGRAMCILIĞI II

Doç. Dr. Ercan TOP

ÜNİTE

7



GİRİŞ

CSS ile ilgili böümlere CSS (Cascading Style Sheets- Basamaklı Stil Sayfaları) kavramından bahsedilerek başlanmıştır. CSS ile ilgili ilk bölümde stil sayfalarının oluşturulduğu kural yapıları, CSS sürümleri, stil tanımlama yöntemleri, stil tanımında kullanılan seçici ve bildirim bloku ve seçici türleri hakkında detaylı bilgiler verilmiştir. Bölümün geri kalanında ise metin düzenlemeyle ilgili sık kullanılan özellikler (yazı tipleri, yazı tipi aileleri, yazı tipi boyutu, metin hizalama, metin dekorasyon, metin ağırlığı, yazı tipi özelliği, metin dönüştürme, bağlantı kullanımı ve satır yüksekliği) ve bu özelliklere atanabilecek değerlerden bahsedilmiştir.

CSS ile ilgili ikinci bölümde ise öncelikle web sayfalarında renk kavramı hakkında bilgiler verilip color, background ve HSLA özelliklerinden ve tanımlanma yöntemlerinden bahsedilmiştir. Span özelliği ile ilgili bilgiler ve kullanımından örnekler verilmiştir. Sonrasında ise tabloların stillerinin düzenlenmesi ile ilgili olarak border, border-collapse, height, width, padding, margin, table color, hover ve nth-child özelliğinden bahsedilmiştir.

Bu bölümde ise liste ile ilgili özelliklerin (list-style-type, list-style-image, list-style position, li:hover) ve görüntülerle ilgili özelliklerin (border, border-image, border-radius, opacity, width, height, background-image) yaygın olarak kullanılanları incelenecaktır. Özelliklerin kullanım şekilleri ve yöntemleri hakkında detaylı açıklamalar yapılacak, kullanım şekilleri ve yöntemleri ile ilgili tam HTML dosyaları veya HTML dosyalarının bazı böümlerinin ekran görüntüleri paylaşılacaktır. Hazırlanan HTML kodlarının tarayıcılarda açıldığında ortaya çıkan görüntüleri de HTML dosyalarının içerikleriyle birlikte verilerek, öğrencilerin HTML kodu ve bu kodlarla ortaya çıkacak görüntüyü zihinlerinde eşleştirmelerine yardımcı olunmaya çalışılacaktır.

LİSTELER

HTML'de iki ana tür liste vardır:

- (unordered list - sıralanmamış listeler): Liste öğeleri imlerle işaretlenir.
- (orderd list - sıralı listeler): Liste öğeleri sayılarla veya harflerle işaretlenir.


CSS list-style-type özelliği, liste maddesi işaretçisinin stilini tanımlamak için kullanılır.

CSS liste özellikleri şunların yapılmasına izin verir:

- Sıralı listeler için farklı liste öğesi işaretleyicileri ayarlama,
- Sırasız listeler için farklı liste öğesi işaretleyicileri ayarlama,
- Bir resmi liste öğesi işaretçisi olarak ayarlama,
- Listelere ve listelemeye öğelerine arka plan renkleri ekleme.

list-style-type

Sıralı ve sıralanmamış listeler için çeşitli seçenekler ve işaretçiler bulunmaktadır. list-style-type özelliği, liste öğesi işaretçisinin türünü belirtir. CSS list-style-type özelliği, liste maddesi işaretçisinin stilini tanımlamak için

kullanılır. Sıralanmamış listeler için list-style-type özelliğinin alabileceği değerler ve açıklamaları aşağıdadır.

- *Disc*: Liste öğesi işaretleyicisini bir madde imine ayarlar (varsayılan).
- *Circle*: Liste öğesi işaretcisini bir daireye ayarlar.
- *Square*: Liste öğesi işaretleyicisini kareye ayarlar.
- *None*: Liste öğeleri işaretlenmez.



Sıralanmamış listeler için list-style-type özelliğinin alabileceği değerler; disc, circle, square ve none olarak belirlenmiştir.

Tablo 7.1.'de sıralanmamış liste için iki tane class bildirimi ve bildirilen bir class tanımının uygulaması görülmektedir. Dosyada "i" ve "ii" adında iki tane class tanımlanmış, her iki class tanımında list-style-type özelliği için farklı özellikler bildirilmiştir. Tanımlanan bu class seçicisinden bir tanesi sıralanmamış listeye uygulanmıştır. Tablo 7.1.'de ayrıca dosyanın tarayıcıda açıldığında görünümü de bulunmaktadır.

Tablo 7.1. Sıralanmamış Liste list-style-type Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul.i{list-style-type: circle;} ul.ii{list-style-type: square;} </style> </head> <body> <ul class="ii"> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body> </html></pre>	<p>■ Atatürk Üniversitesi ■ Açık Öğretim Fakültesi ■ Erzurum</p>

Sıralanmış listeler için list-style-type özelliğinin alabileceği bazı değerler ve açıklamaları aşağıdadır.

- *decimal*: 1 ile başlayan ondalık sayılar
- *decimal-leading-zero*: 10'dan küçük sayılar için önüne 0 eklenir, 1 ile başlayan ondalık sayılar
- *lower-roman*: küçük Romen rakamları
- *upper-roman*: Büyük harf Romen rakamları
- *lower-greek*: Küçük harf Yunanca
- *lower-alpha*: Küçük ASCII harfleri

- *lower-latin*: Küçük ASCII harfleri (IE7'de desteklenmez)
- *upper-alpha*: Büyük ASCII harfleri
- *upper-latin*: Büyük ASCII harfleri (IE7'de desteklenmiyor)
- *georgian*: Gürcü numaralandırma

Tablo 7.2.'de sıralanmış liste için iki tane class bildirimi ve bildirilen bir class tanımının uygulaması görülmektedir. Dosyada "i" ve "ii" adında iki tane class tanımlanmış, her iki class tanımında list-style-type özelliği için farklı özellikler bildirilmiştir. Tanımlanan bu class seçiminden bir tanesi sıralanmış listeye uygulanmıştır. Şekilde ayrıca dosyanın tarayıcıda açıldığındaki görünümü de bulunmaktadır.



list-style-type:none
bildirimi, madde işaretlerini / madde imlerini kaldırmak için kullanılabilir.

Tablo 7.2. Sıralanmış Liste list-style-type Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ol.i{list-style-type: lower-alpha;} ol.ii{list-style-type: decimal-leading-zero;} </style> </head> <body> <ol class="ii"> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body> </html></pre>	<p>01. Atatürk Üniversitesi 02. Açık Öğretim Fakültesi 03. Erzurum</p>

list-style-type:none bildirimi, madde işaretlerini / madde imlerini kaldırmak için kullanılabilir. Ayrıca listenin varsayılan kenar boşluğu ve dolgusu da kaldırılmak istenirse Tablo 7.3.'teki gibi tanımlama yapılabilir. Tablo 7.3.'teki tanımlama ile sıralanmamış listenin madde imleri kaldırılmış ve kenar boşluğu ve dolgu değerleri de sıfır olarak bildirilmiştir. Tablo 7.3.'te ayrıca dosyanın tarayıcıda açıldığındaki gösterimi de bulunmaktadır.

Tablo 7.3. Sıralanmamış Liste list-style-type:none Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul{list-style-type: none; margin:0px; padding:0px;} </style> </head> <body> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body> </html></pre>	<p style="text-align: center;">Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum</p>

list-style-image



list-style-image özelliği, sıralanmamış listeler için liste ögesi işaretçisi olarak bir görüntü tanımlamaya olanak sağlar.

Sıralanmamış listeler için *liste ögesi işaretçisi olarak bir görüntü tanımlamaya* olanak sağlar. Tablo 7.4.'te işaretçi olarak "fakulte.gif" dosyasının işaretçi olarak tanımlanması ve html dosyasının tarayıcıda açıldığı esnada göründüğü de verilmektedir. Tarayıcıdaki görüntüde işaretçi olarak resim dosyası gözlenebilmektedir.

Tablo 7.4. Sıralanmamış Liste list-style-image Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><html><head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul{list-style-image: url("fakulte.gif");} </style></head> <body> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body> </html></pre>	 <p style="text-align: center;">Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum</p>

```
<li> Erzurum </li>
</ul>
</body>
</html>
```

list-style-position

İşaretçilerin konumlandırılmasına olanak sağlar. Bu özellik *İçerde (inside)* ve *dışarıda (outside)* değerleri bildirilerek kullanılabilir. Tablo 7.5.'te inside ve outside özelliğinin sıralanmamış liste için class olarak tanımlanması gösterilmiştir. Tablo 7.5.'te ayrıca bu iki değerin css tanımı yapılan dosyanın web tarayıcısında açıldığındaki görüntüsü de gösterilmektedir.



list-style-position
özellik, işaretçilerin
konumlandırılmasına
olanak sağlar.

Tablo 7.5. Sıralanmamış Liste list-style-type Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><!DOCTYPE html> <html lang="tr"> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul.ic{list-style-position: inside;} ul.dis{list-style-position: outside;} </style> </head> <body> <p>İçerde (inside)</p> <ul class="ic"> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum <p>Dışarda (outside)</p> <ul class="dis"> Atatürk Üniversitesi Açık Öğretim Fakültesi </body></pre>	<p>İçerde (inside) kullanımı</p> <ul style="list-style-type: none"> • Atatürk Üniversitesi • Açık Öğretim Fakültesi • Erzurum <p>Dışarda (outside) kullanımı</p> <ul style="list-style-type: none"> • Atatürk Üniversitesi • Açık Öğretim Fakültesi

</html>	
---------	--

li:hover

Fare üzerine getirildiğinde, farenin üzerinde olduğu liste öğesini vurgulamayı sağlar. Örnek olarak imleç, sıralanmamış listede liste öğesinin üzerine geldiğinde ögenin arka plan rengini değiştiren stil bildirimi “*li:hover{background:#88ee55;}*” olarak tanımlanabilir. Tablo 7.6.’da ** ve ** etiketleri için çeşitli bildirimler ve bütün bildirimlerin sonuçlarının tarayıcıda gösterimi yapılmıştır. Stil tanımlanmasında ** etiketi arka plan rengi ve dolgusu (padding) için değer; ** etiketi arka plan rengi, dolgu ve sol kenar boşluğu için değer; ** etiketinin hover seçicisi arka plan rengi için değer bildirimi yapılmıştır. Tablo 7.6.’daki görüntü web tarayıcısında açıldığında ve imleç Erzurum yazısının üzerinde iken elde edilmiştir. İmleç Erzurum metninin üzerinde iken, stil bildiriminde tanımlandığı gibi diğer sıralanmamış liste öğelerinin arka plan renginden farklı bir renkte arka plan rengi gösterilmiştir.

Tablo 7.6. li:hover Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul { background: #33eeff; padding: 20px;} li { background: #ffee55; padding: 5px; margin-left: 35px;} li:hover { background: #88ee55;} </style> </head> <body> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body></html></pre>	

GÖRSELLER (IMAGES)

Görüntüler (images)



Tüm görsellere stiller ayarlayarak görseller için standart bir görünüm yaratılabilir.

CSS, sayfalarda kullanılan görüntülerin nasıl görüntüleneceğinin ayarlanması olanak sağlar. Tüm görsellere stiller ayarlayarak görseller için standart bir görünüm yaratılabilir. Bu sayede, sayfalar arasında tutarlık sağlanarak sayfa ziyaretçilere daha profesyonel bir izlenim yaratılabilir. Tarayıcıda görüntülerin kenarlıkları (*border*), *border-radius* (oval köşeler), genişlikleri (*width*) / yükseklikleri (*height*), opaklı dereceleri (*opacity*), konumlarını (*background-position*) ve imleç üzerlerine geldiğinde (*hover*) stilleri değiştirmek için CSS tanımlamaları kullanılabilir.

border

Görüntülere kenarlık eklemek için border özelliği kullanılabilir. border özelliği ile kenar boşlukları ve/veya dolgu özelliği kullanmak görüntülerin sayfalarda daha etkili bir şekilde yerleştirilmesi için işe yarayabilmektedir. Tablo 7.7.'de görüntüye ince siyah bir kenarlık ile kenarlık ve görüntü arasına biraz (5px) dolgu ekleyen stil bildirimi gösterilmektedir. Tablo 7.7.'de ayrıca stil tanımı yapılan dosyanın web tarayıcısında açıldığında görüntüsü de gösterilmektedir. Görüntüde görselin etrafına ince düz siyah bir çerçeve çizildiği ve görselin çizilen çerçevenin kenarlarından biraz içerde (padding:5px; bildiriminden dolayı) yerleştirildiği gözlenebilmektedir.

Tablo 7.7. Kenarlık (border) Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre data-bbox="414 1255 997 1810"><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img { border: 1px solid black; padding: 5px; } </style> </head> <body> </body> </html></pre>	

Tablo 7.7.'deki stil bildiriminde bütün kenarlar için kenarlık ve dolgu *tek bir bildirim* ile yapılmıştır. Kenarlık stil tanımında her bir kenar için ayrı ayrı bildirim de yapılmaktadır. Üst kenarlık için *border-top*, sağ kenarlık için *border-right*, alt kenarlık için *border-bottom* ve sol kenarlık için *border-left* özellikleri kullanılabilmektedir.

border-image



border-image özelliği, bir ögenin çevresindeki normal kenarlık yerine kullanılacak bir resim belirlemeye olanak sağlar.

border-image (kenarlık resmi) özelliği ile bir ögenin etrafında kenarlık olarak kullanılacak bir görsel ayarlanabilmektedir. border-image özelliği, bir ögenin çevresindeki *normal kenarlık yerine kullanılacak bir resim belirlemeye* olanak sağlamaktadır. Özellikle tanımlaması *kenarlık olarak kullanılacak görüntü, resmin nereden dilimleneceği* ve *orta bölümlerin tekrarlanacağını veya uzatılacağını tanımlayan* üç bölümden oluşmaktadır. border-image özelliği öncelikle *görüntüyü alıp dokuz kısma* bölmektedir. Sonra *köşeleri köşelere* yerleştirilmektedir ve orta bölümler belirtilen şekilde tekrarlanmakta veya uzatılmaktadır. border-image özelliğinin çalışması için nesnenin *border (kenarlık) özelliğinin* de tanımlanmış olması gerekmektedir. Tablo 7.8.'de border-image özelliği bir class stilinde kullanılmış ve <p> etiketine uygulanmıştır. Border-image tanımında kenarlık için kullanılacak resim (border-image.png Şekil 7.1.'de gösterilmektedir), resmin nasıl dilimleneceği (yüzde 33.3'lük bir oran belirlenmiştir) ve orta bölümün tekrarlanacağı (*round*) belirtilmiştir. Dilimlenen resmin köşeleri border-image özelliği uygulanan alanın köşelerine uygulanmıştır. Yapılan işlemin daha iyi anlaşılabilmesi amacıyla dilimleme için kullanılan resmin (Şekil 7.1.) köşeleri farklı renklerle boyanmıştır. border-image stil tanımının üçüncü bölümü için kullanılabilecek değerler *round* (çevrele), *repeat* (tekrarla) ve *stretch* (uzat) olarak tanımlanmıştır.

Tablo 7.8. Kenarlık Resmi (border-image) Özelliği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <title>Stiller </titl <style> .kenarlik { height:50px; width:50px; padding: 15px; border: 10px solid transparent; border-image: url(border-image.png) 33.3% round;} </style> </head> <body> <p class = "kenarlik">atauni</> </body> </html></pre>	



Şekil 7.1. Kenarlık Olarak Kullanılacak Resim (border-image.png)



border-radius özelliği, öğelere yuvarlatılmış köşeler eklemeye olanak sağlar.

border-radius

Yuvarlak (oval) köşeli görüntüler oluşturmak için border-radius özelliği kullanılabilir. Kenarlık yarıçapı özelliği, ögenin köşelerinin yarıçapını tanımlar. Bir başka ifade ile bu özellik, *ögelere yuvarlatılmış köşeler eklemeye olanak sağlar*. Bütün köşeler için tek değer tanımlanıldığı gibi ayrı ayrı her bir köşe için de yarıçap tanımlanabilmektedir. Tablo 7.9.'da kenar çizgisi olarak bir piksel, düz ve siyah çizgi ve bütün köşeler için tek bir border-radius değeri tanımlaması yapılmıştır. Aynı zamanda Tablo 7.9.'da HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir. Resim dosyasının tarayıcısındaki görüntüsü incelendiğinde görüntünün kenarlarının tam kare olmadığı, kenarlarının yuvarlatılarak gösterildiği gözlemlenebilir.

Tablo 7.9. Yuvarlak Köşeli Görseller (border-radius) Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img {border: 1px solid black; border-radius:20px} </style> <body> </body> </html></pre>	

Aynı görselin *farklı köşeleri için farklı yuvarlama değerleri* de tanımlanabilmektedir. Tablo 7.10.'da border-radius bildiriminde dört farklı değer tanımlanmıştır. İlk değer sol üst köşe (60px), ikinci değer sağ üst köşe (40px), üçüncü değer sağ alt köşe (20px) ve dördüncü değer sol alt köşe (0px) için



border-radius özelliği ile aynı görselin farklı köşeleri için farklı yuvarlama değerleri de tanımlanabilmektedir.

belirlenmiştir. Farklı değerler kullanılarak tanımlanan ve uygulanan stilin tarayıcıda görüntüsü de Tablo 7.10.'da incelenmektedir. Sol üst köşedeki köşenin yarıçapı, sağ üst ve sağ alt köşelerinin yarıçapından ve hiç yuvarlak köşe tanımlanmayan sol alt köşeden açıkça farklı görülmektedir.

Tablo 7.10. Farklı Değerde Yuvarlak Köşeli Görseller Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img {border: 1px solid black; border-radius:60px 40px 20px 0px;} </style> </head> <body> </body> </html></pre>	



border-radius özelliği için yuvarlama değeri olarak % oranı da kullanılabilmektedir.

border-radius bildirimi ile sadece bütün köşelere tek bir değer ve her bir köşe için farklı değer bildirimi yapılmamaktadır. **Border-radius özelliği bildiriminde iki ve üç değer de kullanılabilmektedir.** İki değerli border-radius bildiriminde (örneğin, border-radius {40x 5px;}); ilk değer (60px) sol üst ve sağ alt köşeye, ikinci değer (5px) sağ üst ve sol alt köşelere uygulanır. Üç değerli border-radius bildiriminde (örneğin, border-radius {60x 40px 5px;}); ilk değer (60px) sol üst köşeye uygulanır, ikinci değer (40px) sağ üst ve sol alt köşelere uygulanır ve üçüncü değer (5px) sağ alt köşeye uygulanır.

border-radius özelliği için **değer olarak % oranı** da kullanılabilmektedir. Sayısal değerlerde olduğu gibi bütün köşeler için tek bir yüzdelik değer, her bir köşe için farklı yüzdelik değer, üçlü yüzdelik değer (ilk değer sol üst köşe için, ikinci değer sağ üst ve sol alt köşe için ve üçüncü değer sağ alt köşe için) ve ikili yüzdelik değer (ilk değer sol üst ve sağ alt köşe için, ikinci değer sağ üst ve sol alt köşe için) tanımlanabilmektedir. Tablo 7.11.'de bütün köşeler için tek bir yüzdelik border-radius ve border değer tanımlaması yapılmıştır. Tablo 7.11.'de aynı zamanda HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir. Görsel incelendiğinde, bütün köşelerin aynı oranda ovalleştirildiği gözlenebilmektedir.

Tablo 7.11. Yüzdelik Değerlerle Yuvarlak Köşeli Görsteller Tanımlama Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre data-bbox="403 300 727 1096"><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img {border: 1px solid black; border-radius: 10%;} </style> <body> </body> </html></pre>	

border-radius özelliğinin *yüzdelik değeri 100% olarak tanımlandığında, tam yuvarlak* (görsel tam kare ise daire, görsel tam kare değilse elips) bir görüntü elde edilebilir. `` etiketi stil tanımını `"img {border: 1px solid black; border-radius:100%;}"` olarak yaptığımızda Şekil 7.2.'deki gibi tam elips bir görüntü elde edilebilir. İşlemde kullanılan görsel kare olsaydı görüntü de tam bir daire olacaktı.

**Şekil 7.2.** Tam Oval (border-radius) Örneği

opacity

opacity özelliği kullanarak bir görselin saydamlık derecesini tanımlamabilen opacity özelliği için *0.0 ile 1.0 arasında* değerler tanımlanabilir. Opaklık özelliğinde

 Opacity özelliği bir görselin saydamlık derecesini tanımlamak için kullanılır.

kullanılan değerlerden, *1 görselin hiç şeffaf olmadığını, 0,5 görselin %50 şeffaf olduğunu* ve *0 görselin tamamen şeffaf olduğunu* tanımlar. Tablo 7.12'de `` etiketi için opacity değerleri 0.3, 0.6 ve 1 olan üç class stili tanımlanmıştır. Tanımlanan class stillerinin uygulandığı görsellerin tarayıcıda açılmış hâli incelendiğinde (Şekil 7.3.), farklı opaklık değerlerinin etkisi de kolayca gözle görülmektedir. Opaklık değeri olarak 0.3 kullanılan ilk görselin en saydam olduğu, opaklık değeri olarak 0.6 kullanılan ikinci görselin daha az saydam olduğu ve opaklık değeri olarak 1 kullanılan üçüncü görselde saydamlığın hiç kullanılmadığı görülebilmektedir.

Tablo 7.12. Opaklık (opacity) Örneği

Html Sayfası
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img.bir {opacity: 0.3; img.iki {opacity: 0.6;} img.uc {opacity: 1;} </style> <body> </body> </html></pre>



Şekil 7.3. Farklı Opaklık Değerini Gösterimi

Boyutlar

Görüntülerin yüksekliğini (height) ve genişliğini (width) değiştirmek veya ayarlamak için stil tanımlanabilmektedir. İndirme hızları nedeniyle görüntü boyutlarını ayarlamak için tarayıcıyı kullanmak iyi bir fikir değildir ama yine de sayfaların düzenli görüntülenmesi için sıkılıkla kullanılmaktadır. CSS ile görüntüler standart bir genişlik veya yükseklik değeri kullanılarak ayarlanabilir, hatta boyutların içinde bulunan alana göre otomatik olarak ayarlanması da yapılabilir.



Görüntüler yeniden boyutlandırıldığında, en iyi sonuçları elde etmek için, sadece bir boyutu (yükseklik veya genişlik) yeniden boyutlandırmak yeterlidir.

Görüntüler yeniden boyutlandırıldığında, *en iyi sonuçları elde etmek için, sadece bir boyutu (yükseklik veya genişlik) yeniden boyutlandırmak* gerekir. Bu, görüntünün en boy oranını korumasını sağlar ve böylece görüntünün tarayıcıda değişik / orantısız görünmesi engellenebilir. *Diger değer otomatik olarak ayarlanmalı ya da tarayıcıya en boy oranını koruması bildirilmelidir.* Tablo 7.13.'te `` etiketi için iki farklı class stili tanımlanmıştır. "Bir" adlı ilk class stili tanımlamasında "height" değeri 193 ve "width" değeri auto (otomatik) olarak bildirilmiştir. "İki" adlı ilk class stili tanımlamasında "height" değeri 193 ve "width" değeri 517 olarak bildirilmiştir. Örnekte kullanılan görüntünün (`sosyalmedia.jpg`) boyutu 517*386 pikseldir. Dosya tarayıcıda açıldığında (Şekil 7.4.) "bir" adlı class tanımı uygulanan görselde görüntünün deform olmadığı, sadece boyutunun küçüldüğü görülmektedir. "iki" adlı class tanımı uygulanan görselde görüntünün deform olduğu görülmektedir.

Tablo 7.13. Görsel Boyutlarını Kullanma Örneği

Html Sayfası
<pre> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img.bir {height: 193px; width:auto;} img.iki {height: 193px; width:517px;} </style> </head> <body> </body> </html> </pre>



Web sayfalarında bir arka plan görseli oluşturmak `background-image` özelliği ile oldukça kolay bir işledir.

Şekil 7.4. Dosya Boyutu Web Sayfası Görüntüsü

background-image

CSS, görsellerle süslü arka planlar oluşturmayı kolaylaştırır. Tüm sayfaya, belirli bir bölgeye veya yalnızca belirli bir öğeye arka planlar eklenebilir. Sayfada bir arka plan görseli oluşturmak `background-image` özelliği ile oldukça kolay bir işledir. *Arka plana yalnızca bir görsel yerleştirmek için `<body>` etiketine stil tanımlamak yeterlidir.* Tablo 7.14.'te body etiketinin `background-image` özelliği

için “logo.jpg” dosyası arka plan olarak tanımlanmıştır. Sayfa tarayıcıda açıldığında (Şekil 7.5.) dikeyde ve yatayda “logo.jpg” dosyasının sayfa boyunca yan yana ve alt alta yerleştirildiği görülmektedir. Bu şekilde yapılan tanımla, sayfa boyunca tanımlanan görselin sayfaya yerleşimi yapılmaktadır.

Tablo 7.14. Arka Plan Resim (background-image) Örneği

Html Sayfası
<pre><html> <head> <title>Stiller</title> <style type="text/css"> body {background-image:url(logo.png);} h3 {text-align:center; font-size:50px; color:blue; font-weight:bold; padding:30px; text-transform:uppercase;} </style> </head> <body> <h3>Atatürk Üniversitesi Açık Öğretim Fakültesi</h3> </body> </html></pre>

```

<style type="text/css">
    body {background-image:url(logo.png);}
    h3 {text-align:center; font-size:50px;
        color:blue; font-weight:bold; padding:30px;
        text-transform:uppercase;}
</style>
```



Şekil 7.5. Arka Plan Görseli Web Sayfası Görüntüsü

Görüntülerle yapılabilecek bir diğer özellik de filigran gibi sayfanın geri kalanıyla kaydırma yapmayan bir arka plan görüntüsü oluşturmaktır. “*background-attachment*” özelliği, arka plan görüntüsünün sayfanın geri kalanıyla kaydırılmasını

veya sabitlenmesini belirler. background-attachment” özelliğinin alabileceği değerler;

- *scroll*: Arka plan resmi sayfa ile birlikte içerik kaydırılır. Bu varsayılan değerdir.
- *fixed*: Arka plan resmi sayfa ile birlikte kaymaz. İçerik sabit arka planın üzerinde kayıyor gibi görünür
- *local*: Arka plan resmi, ögenin içeriğiyle birlikte kaydırılır.

Arka plan için diğer bir stil seçeneği *background-repeat* özelliğidir. Bu özellik kullanılarak belirtilen görselin yalnızca yatayda, dikeyde veya sadece bir defa döşenmesi sağlanabilir. Varsayılan olarak, arka plan görüntüsü hem dikey hem de yatay olarak tekrarlanır. Background-repeat özelliğinin alabileceği değerler;

- *repeat*: Arka plan görüntüsünün hem dikey hem de yatay olarak tekrarlandığı varsayılan değerdir. Son görüntü sığmazsa kırılır.
- *repeat-x*: Arka plan görüntüsü sadece yatay olarak tekrarlanır.
- *repeat-y*: Arka plan görüntüsü sadece dikey olarak tekrarlanır.
- *no-repeat*: Arka plan resmi tekrarlanmadı. Resim sadece bir kez gösterilecektir.
- *space*: Arka plan resmi, kırpmaya olmadan mümkün olduğu kadar tekrarlanır. Boşluklar görüntüler arasında eşit olarak dağıtılr.
- *round*: Boşluk doldurmak için arka plan görüntüsü tekrarlanır ve kıvrılır veya uzatılır.

Tablo 7.15.’te <body> etiketine “*background-repeat: repeat-y;*” bildirimi ile “logo.png” dosyasının arka plan olarak sayfa boyunca sadece y ekseninde (yukardan aşağıya) yerleştirileceği tanımlanmıştır. Ayrıca dosyanın tarayıcıda açıldığında ortaya çıkan görüntüsü (Şekil 7.6.) incelendiğinde dosyanın arka plan olarak sadece yukarıdan aşağıya sayfa boyunca yerleştirildiği gösterilmiştir. Stil tanımında kullanılan “*background-attachment: fixed*” özelliği ile arka planın içerikle birlikte kaymaması (arka plan sabit ve yazılar üzerinden akışmış gibi görünmesi) bildirimi yapılmıştır.

Tablo 7.15. Arka Plan Resim (background-image) Örneği

Html Sayfası
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> body {background-image:url(logo.png); background-repeat: repeat-y; background-attachment: fixed;} h3 {text-align:center; font-size:50px; color:blue; font-weight:bold; padding:30px;}</pre>

```

    text-transform:uppercase;}</style>
</head>
<body>
<h3>Atatürk Üniversitesi Açık Öğretim Fakültesi</h3>
</body>
</html>

```



Şekil 7.6. Arka Plan Görüsleri Tekrarlanması Web Sayfası Görüntüsü

Arka plan görüntüsü, sayfalarda `background-position` özelliğine göre yerleştirilir. `Background-position` özelliği ile arka plan görüntüsünün başlangıç konumu ayarlanır. “`background-position`” değeri belirtilmemezse, görüntü her zaman ögenin sol üst köşesine yerleştirilir. `background-position` özelliği için “left top” ve “right center” gibi anahtar kelimeler, sayfanın x ve y koordinatlarının yüzdesi, son olarak da sayfanın x ve y koordinatlarındaki başlangıç noktası değerleri tanımlanabilir. Tablo 7.16.’da “logo.png” dosyasının arka plan görseli olarak sadece bir defa kullanılması “`background-image`” ve “`background-repeat`” özellikleri ile bildirilmiştir. “`background-position:300px 100px;`” bildirimini ile bu tek gösterimin x ekseninde 300. piksel ve y ekseninde 100. piksel kesim noktasından başlaması tanımlanmıştır. Şekil 7.7.’de dosyanın tarayıcıdaki görüntüsü de verilmektedir. Görüntü incelendiğinde görselin sadece bir defa arka plan olarak belirlenen noktaya yerleştirildiği görülmektedir.



`background-position` değeri belirtilmemezse, görüntü her zaman ögenin sol üst köşesine yerleştirilir.

Tablo 7.16. Arka Plan Görüsleri Konum (`background-position`) Örneği

Html Sayfası
<pre> <!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> body {background-image:url(logo.png); background-repeat: no-repeat; background-position:300px 100px; } </pre>

```
h3 {text-align:center; font-size:50px;  
color:blue; font-weight:bold; padding:30px;  
text-transform:uppercase;}  
</style>  
</head>  
<body>  
<h3>Atatürk Üniversitesi Açık Öğretim Fakültesi</h3>  
</body></html>
```

ATATÜRK ÜNİVERSİTESİ AÇIK ÖĞRETİM FAKÜLTESİ

Şekil 7.7. Arka Plan Görseli Yerleşimi Web Sayfası Görüntüsü



Bireysel
Etkinlik

- Görsellere; kenarlık eklemek gibi, css stilleri kullanarak gölgelik eklemek de mümkün müdür? Araştırınız.



Ozet

•LISTS (LİSTELER)

- HTML'de iki ana tür liste vardır:
- (unordered list - sıralanmamış listeler): Liste öğeleri imlerle işaretlenir.
- (ordered list - sıralı listeler): Liste öğeleri sayılarla veya harflerle işaretlenir.
- list-style-type:** Sıralı ve sıralanmamış listeler için çeşitli seçenekler ve işaretçiler bulunmaktadır. List-style-type özelliği, liste ögesi işaretleyicisinin türünü belirtir.
- list-style-image:** Sıralanmamış listeler için liste ögesi işaretçisi olarak bir görüntü tanımlamaya olanak sağlar.
- list-style-position:** İşaretçilerin konumlandırılmasına olanak sağlar. Bu özellik içerisinde (inside) ve dışarıda (outside) değerleri bildirilerek kullanılabilir.
- li:hover:** Fare üzerine getirildiğinde, farenin üzerinde olduğu liste ögesini vurgulamayı sağlar.

•GÖRÜNTÜLER (İMAGES)

- CSS, sayfalarда kullanılan görüntülerin nasıl görüntüleneceğinin ayarlanması olanak sağlar. Tüm görsellere stiller ayarlayarak görseller için standart bir görünüm yaratılabilir. Bu sayede, sayfalar arasında tutarlık sağlanarak sayfa ziyaretçilere daha profesyonel bir izlenim yaratılabilir.
- Border:** Görüntülere kenarlık eklemek için border özelliği kullanılabilir. Border özelliği ile kenar boşlukları ve/veya dolgu özelliği kullanmak görüntülerin sayfalarда daha etkili bir şekilde yerleştirilmesi için işe yarayabilmektedir.
- border-image:** border-image (kenarlık resmi) özelliği ile, bir ögenin etrafındaki kenarlık olarak kullanılacak bir görüntü ayarlanabilmektedir. border-image özelliği, bir elemanın çevresindeki normal kenarlık yerine kullanılacak bir resim belirlemeye olanak sağlamaktadır.
- border-radius:** Yuvarlak (oval) köşeli görüntüler oluşturmak için border-radius özelliği kullanılabilir. Kenarlık yarıçapı özelliği, ögenin köşelerinin yarıçapını tanımlar. Aynı görselin farklı köşeleri için farklı yuvarlama değerleri de tanımlanabilmektedir. Border-radius özelliği bildiriminde bir, iki, üç ve dört değer de kullanılabilmektedir. border-radius özelliğinin yüzdelik değeri 100% olarak tanımlandığında, tam yuvarlak veya elips bir görüntü elde edilebilir.
- Opacity:** opacity özelliği kullanarak bir görüntünün saydamlık derecesi tanımlanabilir. opacity özelliği için 0.0 ile 1.0 arasında değerler tanımlanabilir. Opaklık özelliğinde kullanılan değerlerden, 1 görselin hiç şeffaf olmadığını, 0,5 görselin %50 şeffaf olduğunu ve 0 görselin tamamen şeffaf olduğunu tanımlar.
- Boyutlar:** Görüntülerin yüksekliğini (height) ve genişliğini (width) değiştirmek veya ayarlamak için stil tanımlanabilmektedir. İndirme hızları nedeniyle görüntü boyutlarını ayarlamak için tarayıcıyı kullanmak iyi bir fikir değildir ama yine de sayfaların düzenli görüntülenmesi için sıkılıkla kullanılmaktadır.
- background-image:** CSS, görsellerle süslü arka planlar oluşturmayı kolaylaştırır. Tüm sayfaya veya yalnızca belirli bir öğeye arka planlar eklenebilir. background-attachment" özelliği, arka plan görüntüsünün sayfanın geri kalıyla kaydırılmasını veya sabitlenmesini belirler. background-repeat özelliği kullanılarak belirtilen görselin yalnızca yatayda, dikeyde veya sadece bir defa döşenmesi sağlanabilir.

DEĞERLENDİRME SORULARI

1. Aşağıdakilerden hangisi sıralanmamış listelerdeki list-style-type özelliğinin alabileceği değerlerden değildir?
 - a) disc
 - b) square
 - c) circle
 - d) none
 - e) diamond
2. İmleç, liste öğelerinin üzerinde iken liste ögesinin arka planı aşağıdaki görüntülerdeki gibi olmaktadır. Bu işlemin gerçekleşmesini aşağıdaki hangi stil tanımı gerçekleştirebilir?

<ul style="list-style-type: none">• Atatürk Üniversitesi• Açık Öğretim Fakültesi• Erzurum	<ul style="list-style-type: none">• Atatürk Üniversitesi• Açık Öğretim Fakültesi• Erzurum
---	---

- a)

```
<style type="text/css"> li { padding: 5px; margin-left: 35px;}  
ul{padding:20px;} li:hover { background: #88ee55; width:100px;}  
</style>
```
- b)

```
<style type="text/css"> li { padding: 5px; margin-left: 35px;}  
ul { padding: 20px;} li:hover { background: #88ee55; } </style>
```
- c)

```
<style type="text/css"> li { padding: 5px; margin-left: 35px;}  
ul { padding: 20px;} li:hover { background: #88ee55; height:100px;}  
</style>
```
- d)

```
<style type="text/css"> li { padding: 5px; margin-left: 35px;}  
ul { padding: 20px;} ul:hover { background: #88ee55;} </style>
```
- e)

```
<style type="text/css"> li { background: #88ee55; width:100px; }  
ul { padding: 20px;} li:hover { padding: 5px; margin-left: 35px;}  
</style>
```

Müşteri tatmini ile kârların artışını sağlamak

3. Aşağıdaki stil tanımı uygulanmış liste öğeleri nasıl görünüyor olabilir?

```
<style type="text/css"> li{width:200px; height:auto; background-color:  
aqua;} ul {list-style-type: circle; border:1px dashed coral;} </style>
```

a)

- Atatürk Üniversitesi
- Açık Öğretim Fakültesi
- Erzurum

b)

- Atatürk Üniversitesi
- Açık Öğretim Fakültesi
- Erzurum

c)

- Atatürk Üniversitesi
- Açık Öğretim Fakültesi
- Erzurum

d)

- 01. Atatürk Üniversitesi
- 02. Açık Öğretim Fakültesi
- 03. Erzurum

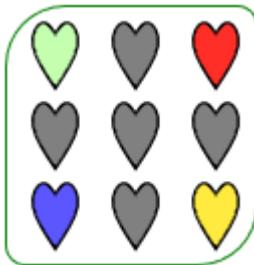
e)

- 01. Atatürk Üniversitesi
- 02. Açık Öğretim Fakültesi
- 03. Erzurum

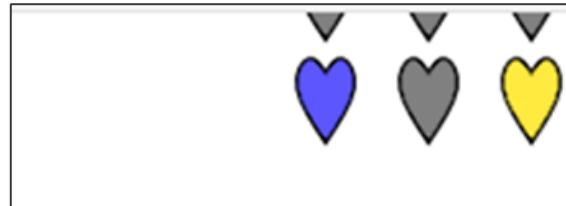
4. Görsellerin css stilleriyle düzenlenmesi ile ilgili aşağıdaki ifadelerden hangisi söylenemez?

- a) Görseller yeniden boyutlandırıldığında, görüntünün bozulmaması için sadede bir boyutu tanımlamak yeterli olabilir.
- b) Görsellerin standart bir şekilde görüntülenebilmesi için stiller kullanılabilir.
- c) "background-attachment" özelliği ile arka plan görüntüsünün sayfanın geri kalanıyla kaydırılması veya sabitlenmesi belirlenir.
- d) "background-repeat" özelliği ile belirtilen görselin ekrana döşenme yönü ve sayısı ile ilgili bilgiler düzenlenenebilir.
- e) Border-radius özelliği ile tam bir daire şekli elde etmek mümkün değildir.

5. Aşağıdaki görüntünün gerçekleşmesini hangi stil tanımı gerçekleştirebilir?



- a) `img{border-image:url("border-image.png") 33% round; border:1px double green;}`
 - b) `img{border-radius:30px 10px; border:1px solid green;}`
 - c) `img{border:30px 10px; background-image:url("border-image.png");}`
 - d) `img{border:10px solid red; background-image:url("border-image.png");}`
 - e) `img{border-radius:30px; border:1px solid green;}`
6. "border-image.png" dosyasının sayfada arka plan resmi olarak aşağıdaki gibi görüntülenmesini hangi css kod bloku sağlayabilir?



border-image.png

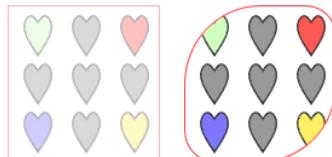
web sayfası görüntüsü

- a) `body {background-image:url(border-image.png); background-repeat:no-repeat; background-position:300px -70px;}`
- b) `img {background-image:url(border-image.png); background-repeat: no-repeat; background-position:300px -70px; }`
- c) `border-radius {background-image:url(border-image.png); background-repeat: no-repeat; background-position:300px -70px; }`
- d) `opacity {background-image:url(border-image.png); background-repeat: no-repeat; background-position:300px -70px; }`
- e) `body {background-image:url(border-image.png); background-repeat: no-repeat; background-position:-70px 300px; }`

7. Aşağıdaki stil tanımları aynı resme ayrı ayrı uygulanıyor. Bu sayfanın görüntüsü hangisi olabilir?

```
<style type="text/css">  
    .bir {opacity:0.3; border:1px solid red;}  
    .iki {border-radius:20% 50%; opacity:0.8; border:1px solid red;}  
</style>
```

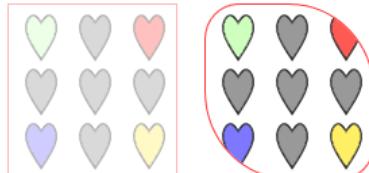
a)



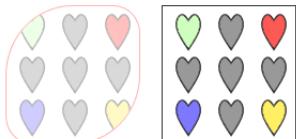
b)



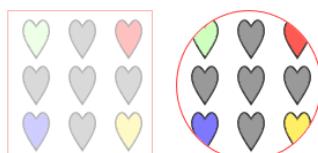
c)



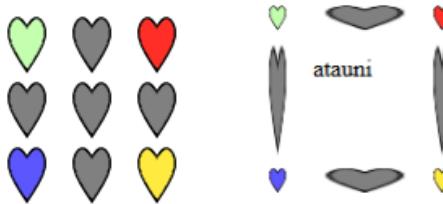
d)



e)



8. Aşağıdaki “border-image.png” dosyası kullanılarak “atauni” metninin web sayfası görüntüsündeki gibi görünmesi sağlanmıştır. Bu işlemin gerçekleşmesini aşağıdaki stil tanımı gerçekleştirebilir?



border-image.png web sayfası görüntüsü

- a) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border: 20px solid transparent; border-image: url(border-image.png) 33.3% round; } </style>`
- b) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border: 20px solid transparent; background-image: url(border-image.png); } </style>`
- c) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border-image: url(border-image.png) 33.3% stretch; } </style>`
- d) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border: 20px solid transparent; border-image: url(border-image.png) 33.3% stretch; } </style>`
- e) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border: 20px solid transparent; border-image: url(border-image.png) 33.3% repeat; } </style>`

9. Arka plan resmi olarak “arka-plan.jpg” dosyasını kullanan, arka plan resmini yatayda tekrarlayan ve arka plan görüntüsünün sayfanın geri kalanıyla birlikte kaymadığı stil tanımı aşağıdakilerden hangisi olabilir?

- a) `body {background-image:url(arka-plan.jpg); background-repeat: repeat-y; background-attachment: fixed;}`
- b) `body {border-image:url(arka-plan.jpg); background-repeat: repeat-x; background-attachment: fixed;}`
- c) `body {background-image:url(arka-plan.jpg); background-repeat: repeat-x; background-attachment: fixed;}`
- d) `img {background-image:url(arka-plan.jpg); background-repeat: repeat-x; background-attachment: scroll;}`
- e) `body {background-image:url(arka-plan.jpg); background-repeat: repeat-x; background-attachment: scroll;}`

10. Aşağıdaki class stil tanımlarından hangisi, uygulandığı görselin saydamlık değerini %50 ve köşeleri 10 piksel çapında yuvarlamak için uygundur?

- a) `<style type="text/css"> .resim {border-radius:20px; opacity:0.5; </style>`
- b) `<style type="text/css"> img {border-radius:20px; opacity:0.5; border:5px solid green;} </style>`
- c) `<style type="text/css"> img {border-radius:20px; opacity:0.5; </style>`
- d) `<style type="text/css"> .resim {border-radius:20px; opacity:0.5; border:5px solid green;} </style>`
- e) `<style type="text/css"> img {border-radius:20px; opacity:50%; border:5px solid green;} </style>`

Cevap Anahtarı

1.e, 2.a, 3.a, 4.e, 5.b, 6.b, 7.c, 8.d, 9.c, 10.d

YARARLANILAN KAYNAKLAR

- Brown, T. B. (2018). CSS master (2nd Edition). VIC: SitePoint Pty. Ltd.
- Dean, J. (2019). Web programming with HTML5, CSS, and Javascript. MA: Jones & Bartlett Learning, LLC, an Ascend Learning Company.
- Grant, K. J. (2018). CSS in Depth. NY: Manning Publications Co.
- Meyer, E. A. (2018). CSS Pocket Reference (5th Edition). CA: O'Reilly Media, Inc.

LİSTE VE GÖRSELLERDE CSS STİLLERİ

İÇİNDEKİLER

- Listeler
- Görseller
 - images
 - border
 - border-image
 - border-radius
 - opacity
 - boyutlar
 - background-image

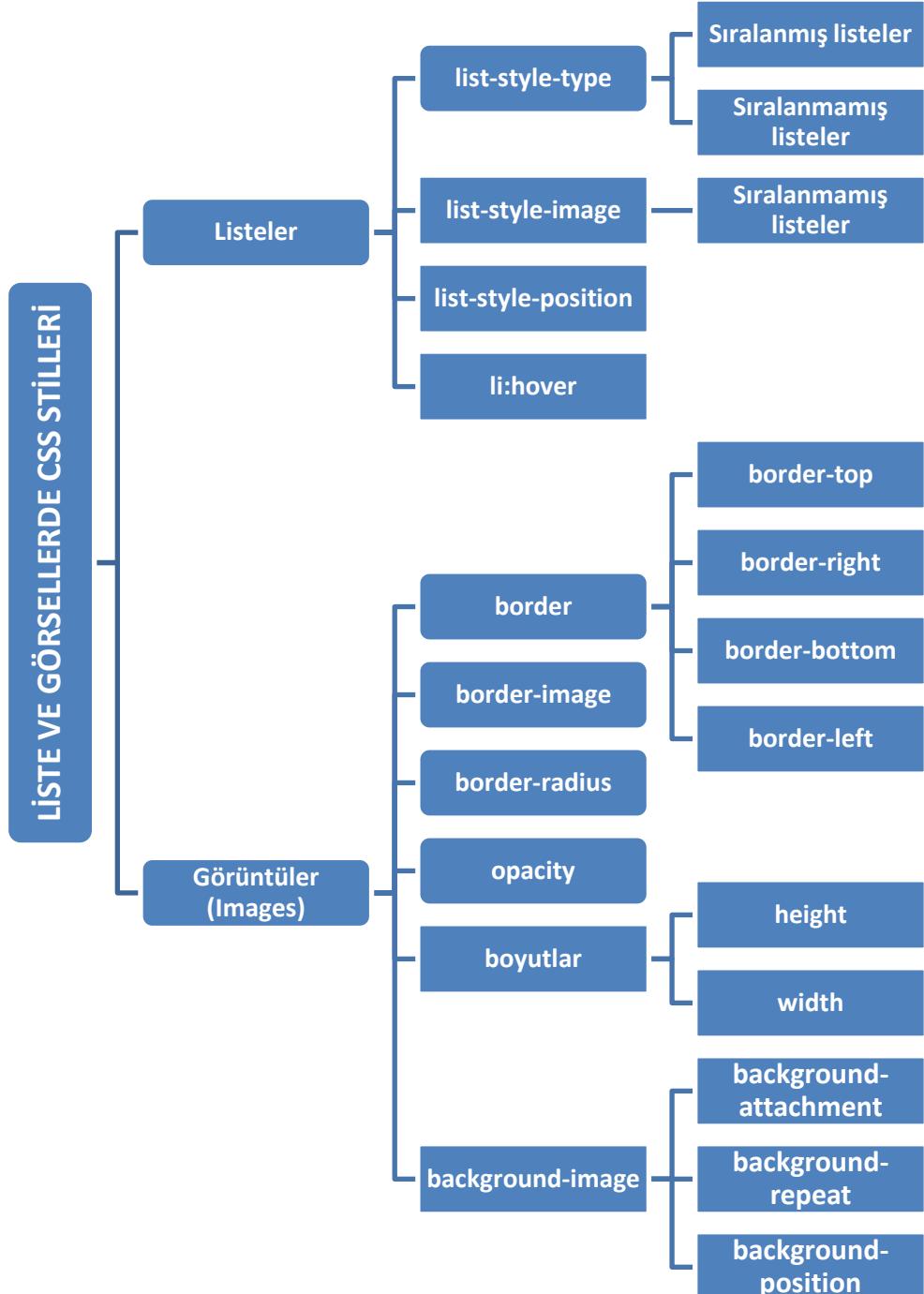
HEDEFLER

- Bu üniteyi çalıştından sonra;
- Listelerle ilgili stillerin neler olduğunu söyleyebilecek,
- Listelerle ilgili css bildirimlerinin sonuçlarını seçebilecek,
- Listelerle ilgili görüntüleri oluşturabilecek ve css tanımlarını gösterebilecek,
- Görsellerle ilgili özellikleri açıklayabilecek,
- Görsellerle ilgili css bildirimlerinin sonuçlarını seçebilecek,
- Görsellerle ilgili görüntüleri oluşturabilecek ve css tanımlarını gösterebileceksiniz.



İNTERNET
PROGRAMCILIĞI II
Doç. Dr. Ercan TOP

ÜNİTE
7



GİRİŞ

CSS ile ilgili böümlere CSS (Cascading Style Sheets- Basamaklı Stil Sayfaları) kavramından bahsedilerek başlanmıştır. CSS ile ilgili ilk bölümde stil sayfalarının oluşturulduğu kural yapıları, CSS sürümleri, stil tanımlama yöntemleri, stil tanımında kullanılan seçici ve bildirim bloku ve seçici türleri hakkında detaylı bilgiler verilmiştir. Bölümün geri kalanında ise metin düzenlemeyle ilgili sık kullanılan özellikler (yazı tipleri, yazı tipi aileleri, yazı tipi boyutu, metin hizalama, metin dekorasyon, metin ağırlığı, yazı tipi özelliği, metin dönüştürme, bağlantı kullanımı ve satır yüksekliği) ve bu özelliklere atanabilecek değerlerden bahsedilmiştir.

CSS ile ilgili ikinci bölümde ise öncelikle web sayfalarında renk kavramı hakkında bilgiler verilip color, background ve HSLA özelliklerinden ve tanımlanma yöntemlerinden bahsedilmiştir. Span özelliği ile ilgili bilgiler ve kullanımından örnekler verilmiştir. Sonrasında ise tabloların stillerinin düzenlenmesi ile ilgili olarak border, border-collapse, height, width, padding, margin, table color, hover ve nth-child özelliğinden bahsedilmiştir.

Bu bölümde ise liste ile ilgili özelliklerin (list-style-type, list-style-image, list-style position, li:hover) ve görüntülerle ilgili özelliklerin (border, border-image, border-radius, opacity, width, height, background-image) yaygın olarak kullanılanları incelenecaktır. Özelliklerin kullanım şekilleri ve yöntemleri hakkında detaylı açıklamalar yapılacak, kullanım şekilleri ve yöntemleri ile ilgili tam HTML dosyaları veya HTML dosyalarının bazı böümlerinin ekran görüntüleri paylaşılacaktır. Hazırlanan HTML kodlarının tarayıcılarda açıldığında ortaya çıkan görüntüleri de HTML dosyalarının içerikleriyle birlikte verilerek, öğrencilerin HTML kodu ve bu kodlarla ortaya çıkacak görüntüyü zihinlerinde eşleştirmelerine yardımcı olunmaya çalışılacaktır.

LİSTELER

HTML'de iki ana tür liste vardır:

- (unordered list - sıralanmamış listeler): Liste öğeleri imlerle işaretlenir.
- (orderd list - sıralı listeler): Liste öğeleri sayılarla veya harflerle işaretlenir.



CSS list-style-type özelliği, liste maddesi işaretçisinin stilini tanımlamak için kullanılır.

CSS liste özellikleri şunların yapılmasına izin verir:

- Sıralı listeler için farklı liste öğesi işaretleyicileri ayarlama,
- Sırasız listeler için farklı liste öğesi işaretleyicileri ayarlama,
- Bir resmi liste öğesi işaretçisi olarak ayarlama,
- Listelere ve listelemeye öğelerine arka plan renkleri ekleme.

list-style-type

Sıralı ve sıralanmamış listeler için çeşitli seçenekler ve işaretçiler bulunmaktadır. list-style-type özelliği, liste öğesi işaretçisinin türünü belirtir. CSS list-style-type özelliği, liste maddesi işaretçisinin stilini tanımlamak için

kullanılır. Sıralanmamış listeler için list-style-type özelliğinin alabileceği değerler ve açıklamaları aşağıdadır.

- *Disc*: Liste öğesi işaretleyicisini bir madde imine ayarlar (varsayılan).
- *Circle*: Liste öğesi işaretcisini bir daireye ayarlar.
- *Square*: Liste öğesi işaretleyicisini kareye ayarlar.
- *None*: Liste öğeleri işaretlenmez.



Sıralanmamış listeler için list-style-type özelliğinin alabileceği değerler; disc, circle, square ve none olarak belirlenmiştir.

Tablo 7.1.'de sıralanmamış liste için iki tane class bildirimi ve bildirilen bir class tanımının uygulaması görülmektedir. Dosyada "i" ve "ii" adında iki tane class tanımlanmış, her iki class tanımında list-style-type özelliği için farklı özellikler bildirilmiştir. Tanımlanan bu class seçicisinden bir tanesi sıralanmamış listeye uygulanmıştır. Tablo 7.1.'de ayrıca dosyanın tarayıcıda açıldığında görünümü de bulunmaktadır.

Tablo 7.1. Sıralanmamış Liste list-style-type Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul.i{list-style-type: circle;} ul.ii{list-style-type: square;} </style> </head> <body> <ul class="ii"> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body> </html></pre>	<p>■ Atatürk Üniversitesi ■ Açık Öğretim Fakültesi ■ Erzurum</p>

Sıralanmış listeler için list-style-type özelliğinin alabileceği bazı değerler ve açıklamaları aşağıdadır.

- *decimal*: 1 ile başlayan ondalık sayılar
- *decimal-leading-zero*: 10'dan küçük sayılar için önüne 0 eklenir, 1 ile başlayan ondalık sayılar
- *lower-roman*: küçük Romen rakamları
- *upper-roman*: Büyük harf Romen rakamları
- *lower-greek*: Küçük harf Yunanca
- *lower-alpha*: Küçük ASCII harfleri

- *lower-latin*: Küçük ASCII harfleri (IE7'de desteklenmez)
- *upper-alpha*: Büyük ASCII harfleri
- *upper-latin*: Büyük ASCII harfleri (IE7'de desteklenmiyor)
- *georgian*: Gürcü numaralandırma

Tablo 7.2.'de sıralanmış liste için iki tane class bildirimi ve bildirilen bir class tanımının uygulaması görülmektedir. Dosyada "i" ve "ii" adında iki tane class tanımlanmış, her iki class tanımında list-style-type özelliği için farklı özellikler bildirilmiştir. Tanımlanan bu class seçiminden bir tanesi sıralanmış listeye uygulanmıştır. Şekilde ayrıca dosyanın tarayıcıda açıldığındaki görünümü de bulunmaktadır.



list-style-type:none
bildirimi, madde işaretlerini / madde imlerini kaldırmak için kullanılabilir.

Tablo 7.2. Sıralanmış Liste list-style-type Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ol.i{list-style-type: lower-alpha;} ol.ii{list-style-type: decimal-leading-zero;} </style> </head> <body> <ol class="ii"> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body> </html></pre>	<p>01. Atatürk Üniversitesi 02. Açık Öğretim Fakültesi 03. Erzurum</p>

list-style-type:none bildirimi, madde işaretlerini / madde imlerini kaldırmak için kullanılabilir. Ayrıca listenin varsayılan kenar boşluğu ve dolgusu da kaldırılmak istenirse Tablo 7.3.'teki gibi tanımlama yapılabilir. Tablo 7.3.'teki tanımlama ile sıralanmamış listenin madde imleri kaldırılmış ve kenar boşluğu ve dolgu değerleri de sıfır olarak bildirilmiştir. Tablo 7.3.'te ayrıca dosyanın tarayıcıda açıldığındaki gösterimi de bulunmaktadır.

Tablo 7.3. Sıralanmamış Liste list-style-type:none Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul{list-style-type: none; margin:0px; padding:0px;} </style> </head> <body> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body> </html></pre>	<p style="text-align: center;">Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum</p>

list-style-image



list-style-image özelliği, sıralanmamış listeler için liste ögesi işaretçisi olarak bir görüntü tanımlamaya olanak sağlar.

Sıralanmamış listeler için *liste ögesi işaretçisi olarak bir görüntü tanımlamaya* olanak sağlar. Tablo 7.4.'te işaretçi olarak "fakulte.gif" dosyasının işaretçi olarak tanımlanması ve html dosyasının tarayıcıda açıldığı esnada göründüğü de verilmektedir. Tarayıcıdaki görüntüde işaretçi olarak resim dosyası gözlenebilmektedir.

Tablo 7.4. Sıralanmamış Liste list-style-image Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><html><head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul{list-style-image: url("fakulte.gif");} </style></head> <body> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body> </html></pre>	 <p style="text-align: center;">Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum</p>

```
<li> Erzurum </li>
</ul>
</body>
</html>
```

list-style-position

İşaretçilerin konumlandırılmasına olanak sağlar. Bu özellik *İçerde (inside)* ve *dışarıda (outside)* değerleri bildirilerek kullanılabilir. Tablo 7.5.'te inside ve outside özelliğinin sıralanmamış liste için class olarak tanımlanması gösterilmiştir. Tablo 7.5.'te ayrıca bu iki değerin css tanımı yapılan dosyanın web tarayıcısında açıldığındaki görüntüsü de gösterilmektedir.



list-style-position
özellikleri, işaretçilerin
konumlandırılmasına
olanak sağlar.

Tablo 7.5. Sıralanmamış Liste list-style-type Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><!DOCTYPE html> <html lang="tr"> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul.ic{list-style-position: inside;} ul.dis{list-style-position: outside;} </style> </head> <body> <p>İçerde (inside)</p> <ul class="ic"> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum <p>Dışarda (outside)</p> <ul class="dis"> Atatürk Üniversitesi Açık Öğretim Fakültesi </body></pre>	<p>İçerde (inside) kullanımı</p> <ul style="list-style-type: none"> • Atatürk Üniversitesi • Açık Öğretim Fakültesi • Erzurum <p>Dışarda (outside) kullanımı</p> <ul style="list-style-type: none"> • Atatürk Üniversitesi • Açık Öğretim Fakültesi

</html>	
---------	--

li:hover

Fare üzerine getirildiğinde, farenin üzerinde olduğu liste öğesini vurgulamayı sağlar. Örnek olarak imleç, sıralanmamış listede liste öğesinin üzerine geldiğinde ögenin arka plan rengini değiştiren stil bildirimi “*li:hover{background:#88ee55;}*” olarak tanımlanabilir. Tablo 7.6.’da ** ve ** etiketleri için çeşitli bildirimler ve bütün bildirimlerin sonuçlarının tarayıcıda gösterimi yapılmıştır. Stil tanımlanmasında ** etiketi arka plan rengi ve dolgusu (padding) için değer; ** etiketi arka plan rengi, dolgu ve sol kenar boşluğu için değer; ** etiketinin hover seçicisi arka plan rengi için değer bildirimi yapılmıştır. Tablo 7.6.’daki görüntü web tarayıcısında açıldığında ve imleç Erzurum yazısının üzerinde iken elde edilmiştir. İmleç Erzurum metninin üzerinde iken, stil bildiriminde tanımlandığı gibi diğer sıralanmamış liste öğelerinin arka plan renginden farklı bir renkte arka plan rengi gösterilmiştir.

Tablo 7.6. li:hover Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> ul { background: #33eeff; padding: 20px;} li { background: #ffee55; padding: 5px; margin-left: 35px;} li:hover { background: #88ee55;} </style> </head> <body> Atatürk Üniversitesi Açık Öğretim Fakültesi Erzurum </body></html></pre>	

GÖRSELLER (IMAGES)

Görüntüler (images)



Tüm görsellere stiller ayarlayarak görseller için standart bir görünüm yaratılabilir.

CSS, sayfalarda kullanılan görüntülerin nasıl görüntüleneceğinin ayarlanması olanak sağlar. Tüm görsellere stiller ayarlayarak görseller için standart bir görünüm yaratılabilir. Bu sayede, sayfalar arasında tutarlık sağlanarak sayfa ziyaretçilere daha profesyonel bir izlenim yaratılabilir. Tarayıcıda görüntülerin kenarlıkları (*border*), *border-radius* (oval köşeler), genişlikleri (*width*) / yükseklikleri (*height*), opaklı dereceleri (*opacity*), konumlarını (*background-position*) ve imleç üzerlerine geldiğinde (*hover*) stilleri değiştirmek için CSS tanımlamaları kullanılabilir.

border

Görüntülere kenarlık eklemek için border özelliği kullanılabilir. border özelliği ile kenar boşlukları ve/veya dolgu özelliği kullanmak görüntülerin sayfalarda daha etkili bir şekilde yerleştirilmesi için işe yarayabilmektedir. Tablo 7.7.'de görüntüye ince siyah bir kenarlık ile kenarlık ve görüntü arasına biraz (5px) dolgu ekleyen stil bildirimi gösterilmektedir. Tablo 7.7.'de ayrıca stil tanımı yapılan dosyanın web tarayıcısında açıldığında görüntüsü de gösterilmektedir. Görüntüde görselin etrafına ince düz siyah bir çerçeve çizildiği ve görselin çizilen çerçevenin kenarlarından biraz içerde (padding:5px; bildiriminden dolayı) yerleştirildiği gözlenebilmektedir.

Tablo 7.7. Kenarlık (border) Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre data-bbox="414 1253 997 1810"><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img { border: 1px solid black; padding: 5px; } </style> </head> <body> </body> </html></pre>	

Tablo 7.7.'deki stil bildiriminde bütün kenarlar için kenarlık ve dolgu *tek bir bildirim* ile yapılmıştır. Kenarlık stil tanımında her bir kenar için ayrı ayrı bildirim de yapılmaktadır. Üst kenarlık için *border-top*, sağ kenarlık için *border-right*, alt kenarlık için *border-bottom* ve sol kenarlık için *border-left* özellikleri kullanılabilmektedir.

border-image



border-image özelliği, bir ögenin çevresindeki normal kenarlık yerine kullanılacak bir resim belirlemeye olanak sağlar.

border-image (kenarlık resmi) özelliği ile bir ögenin etrafında kenarlık olarak kullanılacak bir görsel ayarlanabilmektedir. border-image özelliği, bir ögenin çevresindeki *normal kenarlık yerine kullanılacak bir resim belirlemeye* olanak sağlamaktadır. Özellikle tanımlaması *kenarlık olarak kullanılacak görüntü, resmin nereden dilimleneceği* ve *orta bölümlerin tekrarlanacağını veya uzatılacağını tanımlayan* üç bölümden oluşmaktadır. border-image özelliği öncelikle *görüntüyü alıp dokuz kısma* bölmektedir. Sonra *köşeleri köşelere* yerleştirilmektedir ve orta bölümler belirtilen şekilde tekrarlanmakta veya uzatılmaktadır. border-image özelliğinin çalışması için nesnenin *border (kenarlık) özelliğinin* de tanımlanmış olması gerekmektedir. Tablo 7.8.'de border-image özelliği bir class stilinde kullanılmış ve <p> etiketine uygulanmıştır. Border-image tanımında kenarlık için kullanılacak resim (border-image.png Şekil 7.1.'de gösterilmektedir), resmin nasıl dilimleneceği (yüzde 33.3'lük bir oran belirlenmiştir) ve orta bölümün tekrarlanacağı (*round*) belirtilmiştir. Dilimlenen resmin köşeleri border-image özelliği uygulanan alanın köşelerine uygulanmıştır. Yapılan işlemin daha iyi anlaşılabilmesi amacıyla dilimleme için kullanılan resmin (Şekil 7.1.) köşeleri farklı renklerle boyanmıştır. border-image stil tanımının üçüncü bölümü için kullanılabilecek değerler *round* (çevrele), *repeat* (tekrarla) ve *stretch* (uzat) olarak tanımlanmıştır.

Tablo 7.8. Kenarlık Resmi (border-image) Özelliği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <title>Stiller </titl <style> .kenarlik { height:50px; width:50px; padding: 15px; border: 10px solid transparent; border-image: url(border-image.png) 33.3% round;} </style> </head> <body> <p class = "kenarlik">atauni</> </body> </html></pre>	



Şekil 7.1. Kenarlık Olarak Kullanılacak Resim (border-image.png)



border-radius özelliği, öğelere yuvarlatılmış köşeler eklemeye olanak sağlar.

border-radius

Yuvarlak (oval) köşeli görüntüler oluşturmak için border-radius özelliği kullanılabilir. Kenarlık yarıçapı özelliği, ögenin köşelerinin yarıçapını tanımlar. Bir başka ifade ile bu özellik, *ögelere yuvarlatılmış köşeler eklemeye olanak sağlar*. Bütün köşeler için tek değer tanımlanıldığı gibi ayrı ayrı her bir köşe için de yarıçap tanımlanabilmektedir. Tablo 7.9.'da kenar çizgisi olarak bir piksel, düz ve siyah çizgi ve bütün köşeler için tek bir border-radius değeri tanımlaması yapılmıştır. Aynı zamanda Tablo 7.9.'da HTML dosyasının tarayıcıda açılmış hali de gösterilmektedir. Resim dosyasının tarayıcısındaki görüntüsü incelendiğinde görüntünün kenarlarının tam kare olmadığı, kenarlarının yuvarlatılarak gösterildiği gözlemlenebilir.

Tablo 7.9. Yuvarlak Köşeli Görseller (border-radius) Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img {border: 1px solid black; border- radius:20px} </style> <body> </body> </html></pre>	

Aynı görselin *farklı köşeleri için farklı yuvarlama değerleri* de tanımlanabilmektedir. Tablo 7.10.'da border-radius bildiriminde dört farklı değer tanımlanmıştır. İlk değer sol üst köşe (60px), ikinci değer sağ üst köşe (40px), üçüncü değer sağ alt köşe (20px) ve dördüncü değer sol alt köşe (0px) için



border-radius özelliği ile aynı görselin farklı köşeleri için farklı yuvarlama değerleri de tanımlanabilmektedir.

belirlenmiştir. Farklı değerler kullanılarak tanımlanan ve uygulanan stilin tarayıcıda görüntüsü de Tablo 7.10.'da incelenmektedir. Sol üst köşedeki köşenin yarıçapı, sağ üst ve sağ alt köşelerinin yarıçapından ve hiç yuvarlak köşe tanımlanmayan sol alt köşeden açıkça farklı görülmektedir.

Tablo 7.10. Farklı Değerde Yuvarlak Köşeli Görseller Örneği

Html Sayfası	Sayfanın Tarayıcısındaki Görüntüsü
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img {border: 1px solid black; border-radius:60px 40px 20px 0px;} </style> </head> <body> </body> </html></pre>	



border-radius özelliği için yuvarlama değeri olarak % oranı da kullanılabilmektedir.

border-radius bildirimi ile sadece bütün köşelere tek bir değer ve her bir köşe için farklı değer bildirimi yapılmamaktadır. **Border-radius özelliği bildiriminde iki ve üç değer de kullanılabilmektedir.** İki değerli border-radius bildiriminde (örneğin, border-radius {40x 5px;}); ilk değer (60px) sol üst ve sağ alt köşeye, ikinci değer (5px) sağ üst ve sol alt köşelere uygulanır. Üç değerli border-radius bildiriminde (örneğin, border-radius {60x 40px 5px;}); ilk değer (60px) sol üst köşeye uygulanır, ikinci değer (40px) sağ üst ve sol alt köşelere uygulanır ve üçüncü değer (5px) sağ alt köşeye uygulanır.

border-radius özelliği için **değer olarak % oranı** da kullanılabilmektedir. Sayısal değerlerde olduğu gibi bütün köşeler için tek bir yüzdelik değer, her bir köşe için farklı yüzdelik değer, üçlü yüzdelik değer (ilk değer sol üst köşe için, ikinci değer sağ üst ve sol alt köşe için ve üçüncü değer sağ alt köşe için) ve ikili yüzdelik değer (ilk değer sol üst ve sağ alt köşe için, ikinci değer sağ üst ve sol alt köşe için) tanımlanabilmektedir. Tablo 7.11.'de bütün köşeler için tek bir yüzdelik border-radius ve border değer tanımlaması yapılmıştır. Tablo 7.11.'de aynı zamanda HTML dosyasının tarayıcıda açılmış hâli de gösterilmektedir. Görsel incelendiğinde, bütün köşelerin aynı oranda ovalleştirildiği gözlenebilmektedir.

Tablo 7.11. Yüzdelik Değerlerle Yuvarlak Köşeli Görsteller Tanımlama Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre data-bbox="403 300 727 1096"><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img {border: 1px solid black; border-radius: 10%;} </style> <body> </body> </html></pre>	

border-radius özelliğinin *yüzdelik değeri 100% olarak tanımlandığında, tam yuvarlak* (görsel tam kare ise daire, görsel tam kare değilse elips) bir görüntü elde edilebilir. `` etiketi stil tanımını `"img {border: 1px solid black; border-radius:100%;}"` olarak yaptığımızda Şekil 7.2.'deki gibi tam elips bir görüntü elde edilebilir. İşlemde kullanılan görsel kare olsaydı görüntü de tam bir daire olacaktı.

**Şekil 7.2.** Tam Oval (border-radius) Örneği

opacity

opacity özelliği kullanarak bir görselin saydamlık derecesini tanımlamabilen opacity özelliği için *0.0 ile 1.0 arasında* değerler tanımlanabilir. Opaklık özelliğinde

 Opacity özelliği bir görselin saydamlık derecesini tanımlamak için kullanılır.

kullanılan değerlerden, *1 görselin hiç şeffaf olmadığını, 0,5 görselin %50 şeffaf olduğunu* ve *0 görselin tamamen şeffaf olduğunu* tanımlar. Tablo 7.12'de `` etiketi için opacity değerleri 0.3, 0.6 ve 1 olan üç class stili tanımlanmıştır. Tanımlanan class stillerinin uygulandığı görsellerin tarayıcıda açılmış hâli incelendiğinde (Şekil 7.3.), farklı opaklık değerlerinin etkisi de kolayca gözle görülmektedir. Opaklık değeri olarak 0.3 kullanılan ilk görselin en saydam olduğu, opaklık değeri olarak 0.6 kullanılan ikinci görselin daha az saydam olduğu ve opaklık değeri olarak 1 kullanılan üçüncü görselde saydamlığın hiç kullanılmadığı görülebilmektedir.

Tablo 7.12. Opaklık (opacity) Örneği

Html Sayfası
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img.bir {opacity: 0.3; img.iki {opacity: 0.6;} img.uc {opacity: 1;} </style> <body> </body> </html></pre>



Şekil 7.3. Farklı Opaklık Değerini Gösterimi

Boyutlar

Görüntülerin yüksekliğini (height) ve genişliğini (width) değiştirmek veya ayarlamak için stil tanımlanabilmektedir. İndirme hızları nedeniyle görüntü boyutlarını ayarlamak için tarayıcıyı kullanmak iyi bir fikir değildir ama yine de sayfaların düzenli görüntülenmesi için sıkılıkla kullanılmaktadır. CSS ile görüntüler standart bir genişlik veya yükseklik değeri kullanılarak ayarlanabilir, hatta boyutların içinde bulunan alana göre otomatik olarak ayarlanması da yapılabilir.



Görüntüler yeniden boyutlandırıldığında, en iyi sonuçları elde etmek için, sadece bir boyutu (yükseklik veya genişlik) yeniden boyutlandırmak yeterlidir.

Görüntüler yeniden boyutlandırıldığında, *en iyi sonuçları elde etmek için, sadece bir boyutu (yükseklik veya genişlik) yeniden boyutlandırmak* gerekir. Bu, görüntünün en boy oranını korumasını sağlar ve böylece görüntünün tarayıcıda değişik / orantısız görünmesi engellenebilir. *Diger değer otomatik olarak ayarlanmalı ya da tarayıcıya en boy oranını koruması bildirilmelidir.* Tablo 7.13.'te `` etiketi için iki farklı class stili tanımlanmıştır. "Bir" adlı ilk class stili tanımlamasında "height" değeri 193 ve "width" değeri auto (otomatik) olarak bildirilmiştir. "İki" adlı ilk class stili tanımlamasında "height" değeri 193 ve "width" değeri 517 olarak bildirilmiştir. Örnekte kullanılan görüntünün (`sosyalmedia.jpg`) boyutu 517*386 pikseldir. Dosya tarayıcıda açıldığında (Şekil 7.4.) "bir" adlı class tanımı uygulanan görselde görüntünün deform olmadığı, sadece boyutunun küçüldüğü görülmektedir. "iki" adlı class tanımı uygulanan görselde görüntünün deform olduğu görülmektedir.

Tablo 7.13. Görsel Boyutlarını Kullanma Örneği

Html Sayfası
<pre> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> img.bir {height: 193px; width:auto;} img.iki {height: 193px; width:517px;} </style> </head> <body> </body> </html> </pre>



Web sayfalarında bir arka plan görseli oluşturmak `background-image` özelliği ile oldukça kolay bir işledir.

Şekil 7.4. Dosya Boyutu Web Sayfası Görüntüsü

background-image

CSS, görsellerle süslü arka planlar oluşturmayı kolaylaştırır. Tüm sayfaya, belirli bir bölgeye veya yalnızca belirli bir öğeye arka planlar eklenebilir. Sayfada bir arka plan görseli oluşturmak `background-image` özelliği ile oldukça kolay bir işledir. *Arka plana yalnızca bir görsel yerleştirmek için `<body>` etiketine stil tanımlamak yeterlidir.* Tablo 7.14.'te body etiketinin `background-image` özelliği

için “logo.jpg” dosyası arka plan olarak tanımlanmıştır. Sayfa tarayıcıda açıldığında (Şekil 7.5.) dikeyde ve yatayda “logo.jpg” dosyasının sayfa boyunca yan yana ve alt alta yerleştirildiği görülmektedir. Bu şekilde yapılan tanımla, sayfa boyunca tanımlanan görselin sayfaya yerleşimi yapılmaktadır.

Tablo 7.14. Arka Plan Resim (background-image) Örneği

Html Sayfası
<pre><html> <head> <title>Stiller</title> <style type="text/css"> body {background-image:url(logo.png);} h3 {text-align:center; font-size:50px; color:blue; font-weight:bold; padding:30px; text-transform:uppercase;} </style> </head> <body> <h3>Atatürk Üniversitesi Açık Öğretim Fakültesi</h3> </body> </html></pre>

```

<style type="text/css">
    body {background-image:url(logo.png);}
    h3 {text-align:center; font-size:50px;
        color:blue; font-weight:bold; padding:30px;
        text-transform:uppercase;}
</style>
```



Şekil 7.5. Arka Plan Görseli Web Sayfası Görüntüsü

Görüntülerle yapılabilecek bir diğer özellik de filigran gibi sayfanın geri kalanıyla kaydırma yapmayan bir arka plan görüntüsü oluşturmaktır. “*background-attachment*” özelliği, arka plan görüntüsünün sayfanın geri kalanıyla kaydırılmasını

veya sabitlenmesini belirler. background-attachment” özelliğinin alabileceği değerler;

- *scroll*: Arka plan resmi sayfa ile birlikte içerik kaydırılır. Bu varsayılan değerdir.
- *fixed*: Arka plan resmi sayfa ile birlikte kaymaz. İçerik sabit arka planın üzerinde kayıyor gibi görünür
- *local*: Arka plan resmi, ögenin içeriğiyle birlikte kaydırılır.

Arka plan için diğer bir stil seçeneği *background-repeat* özelliğidir. Bu özellik kullanılarak belirtilen görselin yalnızca yatayda, dikeyde veya sadece bir defa döşenmesi sağlanabilir. Varsayılan olarak, arka plan görüntüsü hem dikey hem de yatay olarak tekrarlanır. Background-repeat özelliğinin alabileceği değerler;

- *repeat*: Arka plan görüntüsünün hem dikey hem de yatay olarak tekrarlandığı varsayılan değerdir. Son görüntü sığmazsa kırılır.
- *repeat-x*: Arka plan görüntüsü sadece yatay olarak tekrarlanır.
- *repeat-y*: Arka plan görüntüsü sadece dikey olarak tekrarlanır.
- *no-repeat*: Arka plan resmi tekrarlanmadı. Resim sadece bir kez gösterilecektir.
- *space*: Arka plan resmi, kırpmaya olmadan mümkün olduğu kadar tekrarlanır. Boşluklar görüntüler arasında eşit olarak dağıtılr.
- *round*: Boşluk doldurmak için arka plan görüntüsü tekrarlanır ve kıvrılır veya uzatılır.

Tablo 7.15.’te <body> etiketine “*background-repeat: repeat-y;*” bildirimi ile “logo.png” dosyasının arka plan olarak sayfa boyunca sadece y ekseninde (yukardan aşağıya) yerleştirileceği tanımlanmıştır. Ayrıca dosyanın tarayıcıda açıldığında ortaya çıkan görüntüsü (Şekil 7.6.) incelendiğinde dosyanın arka plan olarak sadece yukarıdan aşağıya sayfa boyunca yerleştirildiği gösterilmiştir. Stil tanımında kullanılan “*background-attachment: fixed*” özelliği ile arka planın içerikle birlikte kaymaması (arka plan sabit ve yazılar üzerinden akışmış gibi görünmesi) bildirimi yapılmıştır.

Tablo 7.15. Arka Plan Resim (background-image) Örneği

Html Sayfası
<pre><!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> body {background-image:url(logo.png); background-repeat: repeat-y; background-attachment: fixed;} h3 {text-align:center; font-size:50px; color:blue; font-weight:bold; padding:30px;}</pre>

```

    text-transform:uppercase;}</style>
</head>
<body>
<h3>Atatürk Üniversitesi Açık Öğretim Fakültesi</h3>
</body>
</html>

```



Şekil 7.6. Arka Plan Görüsleri Tekrarlanması Web Sayfası Görüntüsü

Arka plan görüntüsü, sayfalarda `background-position` özelliğine göre yerleştirilir. `Background-position` özelliği ile arka plan görüntüsünün başlangıç konumu ayarlanır. “`background-position`” değeri belirtilmemezse, görüntü her zaman ögenin sol üst köşesine yerleştirilir. `background-position` özelliği için “left top” ve “right center” gibi anahtar kelimeler, sayfanın x ve y koordinatlarının yüzdesi, son olarak da sayfanın x ve y koordinatlarındaki başlangıç noktası değerleri tanımlanabilir. Tablo 7.16.’da “logo.png” dosyasının arka plan görseli olarak sadece bir defa kullanılması “`background-image`” ve “`background-repeat`” özellikleri ile bildirilmiştir. “`background-position:300px 100px;`” bildirimini ile bu tek göstergemin x ekseninde 300. piksel ve y ekseninde 100. piksel kesim noktasından başlaması tanımlanmıştır. Şekil 7.7.’de dosyanın tarayıcıdaki görüntüsü de verilmektedir. Görüntü incelendiğinde görselin sadece bir defa arka plan olarak belirlenen noktaya yerleştirildiği görülmektedir.



`background-position` değeri belirtilmemezse, görüntü her zaman ögenin sol üst köşesine yerleştirilir.

Tablo 7.16. Arka Plan Görüsleri Konum (`background-position`) Örneği

Html Sayfası
<pre> <!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> body {background-image:url(logo.png); background-repeat: no-repeat; background-position:300px 100px; } </pre>

```
h3 {text-align:center; font-size:50px;  
color:blue; font-weight:bold; padding:30px;  
text-transform:uppercase;}  
</style>  
</head>  
<body>  
<h3>Atatürk Üniversitesi Açık Öğretim Fakültesi</h3>  
</body></html>
```

ATATÜRK ÜNİVERSİTESİ AÇIK ÖĞRETİM FAKÜLTESİ

Şekil 7.7. Arka Plan Görseli Yerleşimi Web Sayfası Görüntüsü



Bireysel
Etkinlik

- Görsellere; kenarlık eklemek gibi, css stilleri kullanarak gölgelik eklemek de mümkün müdür? Araştırınız.



Ozet

•LISTS (LİSTELER)

- HTML'de iki ana tür liste vardır:
- ** (unordered list - sıralanmamış listeler): Liste öğeleri imlerle işaretlenir.
- ** (ordered list - sıralı listeler): Liste öğeleri sayılarla veya harflerle işaretlenir.
- list-style-type:** Sıralı ve sıralanmamış listeler için çeşitli seçenekler ve işaretçiler bulunmaktadır. List-style-type özelliği, liste ögesi işaretleyicisinin türünü belirtir.
- list-style-image:** Sıralanmamış listeler için liste ögesi işaretçisi olarak bir görüntü tanımlamaya olanak sağlar.
- list-style-position:** İşaretçilerin konumlandırılmasına olanak sağlar. Bu özellik içerisinde (inside) ve dışarıda (outside) değerleri bildirilerek kullanılabilir.
- li:hover:** Fare üzerine getirildiğinde, farenin üzerinde olduğu liste ögesini vurgulamayı sağlar.

•GÖRÜNTÜLER (İMAGES)

- CSS, sayfalarда kullanılan görüntülerin nasıl görüntüleneceğinin ayarlanması olanak sağlar. Tüm görsellere stiller ayarlayarak görseller için standart bir görünüm yaratılabilir. Bu sayede, sayfalar arasında tutarlık sağlanarak sayfa ziyaretçilere daha profesyonel bir izlenim yaratılabilir.
- Border:** Görüntülere kenarlık eklemek için border özelliği kullanılabilir. Border özelliği ile kenar boşlukları ve/veya dolgu özelliği kullanmak görüntülerin sayfalarда daha etkili bir şekilde yerleştirilmesi için işe yarayabilmektedir.
- border-image:** border-image (kenarlık resmi) özelliği ile, bir ögenin etrafındaki kenarlık olarak kullanılacak bir görüntü ayarlanabilmektedir. border-image özelliği, bir elemanın çevresindeki normal kenarlık yerine kullanılacak bir resim belirlemeye olanak sağlamaktadır.
- border-radius:** Yuvarlak (oval) köşeli görüntüler oluşturmak için border-radius özelliği kullanılabilir. Kenarlık yarıçapı özelliği, ögenin köşelerinin yarıçapını tanımlar. Aynı görselin farklı köşeleri için farklı yuvarlama değerleri de tanımlanabilmektedir. Border-radius özelliği bildiriminde bir, iki, üç ve dört değer de kullanılabilmektedir. border-radius özelliğinin yüzdelik değeri 100% olarak tanımlandığında, tam yuvarlak veya elips bir görüntü elde edilebilir.
- Opacity:** opacity özelliği kullanarak bir görüntünün saydamlık derecesi tanımlanabilir. opacity özelliği için 0.0 ile 1.0 arasında değerler tanımlanabilir. Opaklık özelliğinde kullanılan değerlerden, 1 görselin hiç şeffaf olmadığını, 0,5 görselin %50 şeffaf olduğunu ve 0 görselin tamamen şeffaf olduğunu tanımlar.
- Boyutlar:** Görüntülerin yüksekliğini (height) ve genişliğini (width) değiştirmek veya ayarlamak için stil tanımlanabilmektedir. İndirme hızları nedeniyle görüntü boyutlarını ayarlamak için tarayıcıyı kullanmak iyi bir fikir değildir ama yine de sayfaların düzenli görüntülenmesi için sıkılıkla kullanılmaktadır.
- background-image:** CSS, görsellerle süslü arka planlar oluşturmayı kolaylaştırır. Tüm sayfaya veya yalnızca belirli bir öğeye arka planlar eklenebilir. background-attachment" özelliği, arka plan görüntüsünün sayfanın geri kalıyla kaydırılmasını veya sabitlenmesini belirler. background-repeat özelliği kullanılarak belirtilen görselin yalnızca yatayda, dikeyde veya sadece bir defa döşenmesi sağlanabilir.

DEĞERLENDİRME SORULARI

1. Aşağıdakilerden hangisi sıralanmamış listelerdeki list-style-type özelliğinin alabileceği değerlerden değildir?
 - a) disc
 - b) square
 - c) circle
 - d) none
 - e) diamond
2. İmleç, liste öğelerinin üzerinde iken liste ögesinin arka planı aşağıdaki görüntülerdeki gibi olmaktadır. Bu işlemin gerçekleşmesini aşağıdaki hangi stil tanımı gerçekleştirebilir?

<ul style="list-style-type: none">• Atatürk Üniversitesi• Açık Öğretim Fakültesi• Erzurum	<ul style="list-style-type: none">• Atatürk Üniversitesi• Açık Öğretim Fakültesi• Erzurum
---	---

- a)

```
<style type="text/css"> li { padding: 5px; margin-left: 35px;}  
ul{padding:20px;} li:hover { background: #88ee55; width:100px;}  
</style>
```
- b)

```
<style type="text/css"> li { padding: 5px; margin-left: 35px;}  
ul { padding: 20px;} li:hover { background: #88ee55; } </style>
```
- c)

```
<style type="text/css"> li { padding: 5px; margin-left: 35px;}  
ul { padding: 20px;} li:hover { background: #88ee55; height:100px;}  
</style>
```
- d)

```
<style type="text/css"> li { padding: 5px; margin-left: 35px;}  
ul { padding: 20px;} ul:hover { background: #88ee55;} </style>
```
- e)

```
<style type="text/css"> li { background: #88ee55; width:100px; }  
ul { padding: 20px;} li:hover { padding: 5px; margin-left: 35px;}  
</style>
```

Müşteri tatmini ile kârların artışını sağlamak

3. Aşağıdaki stil tanımı uygulanmış liste öğeleri nasıl görünüyor olabilir?

```
<style type="text/css"> li{width:200px; height:auto; background-color:  
aqua;} ul {list-style-type: circle; border:1px dashed coral;} </style>
```

a)

- Atatürk Üniversitesi
- Açık Öğretim Fakültesi
- Erzurum

b)

- Atatürk Üniversitesi
- Açık Öğretim Fakültesi
- Erzurum

c)

- Atatürk Üniversitesi
- Açık Öğretim Fakültesi
- Erzurum

d)

- 01. Atatürk Üniversitesi
- 02. Açık Öğretim Fakültesi
- 03. Erzurum

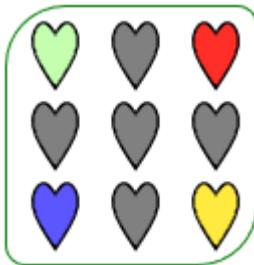
e)

- 01. Atatürk Üniversitesi
- 02. Açık Öğretim Fakültesi
- 03. Erzurum

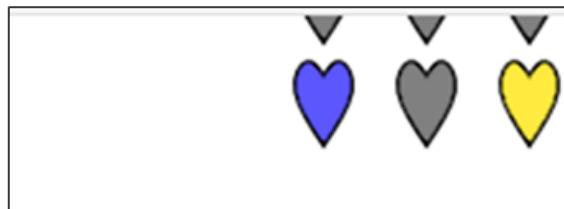
4. Görsellerin css stilleriyle düzenlenmesi ile ilgili aşağıdaki ifadelerden hangisi söylenemez?

- a) Görseller yeniden boyutlandırıldığında, görüntünün bozulmaması için sadede bir boyutu tanımlamak yeterli olabilir.
- b) Görsellerin standart bir şekilde görüntülenebilmesi için stiller kullanılabilir.
- c) "background-attachment" özelliği ile arka plan görüntüsünün sayfanın geri kalanıyla kaydırılması veya sabitlenmesi belirlenir.
- d) "background-repeat" özelliği ile belirtilen görselin ekrana döşenme yönü ve sayısı ile ilgili bilgiler düzenlenenebilir.
- e) Border-radius özelliği ile tam bir daire şekli elde etmek mümkün değildir.

5. Aşağıdaki görüntünün gerçekleşmesini hangi stil tanımı gerçekleştirebilir?



- a) img{border-image:url("border-image.png") 33% round; border:1px double green;}
 - b) img{border-radius:30px 10px; border:1px solid green;}
 - c) img{border:30px 10px; background-image:url("border-image.png");}
 - d) img{border:10px solid red; background-image:url("border-image.png");}
 - e) img{border-radius:30px; border:1px solid green;}
6. "border-image.png" dosyasının sayfada arka plan resmi olarak aşağıdaki gibi görüntülenmesini hangi css kod bloku sağlayabilir?



border-image.png

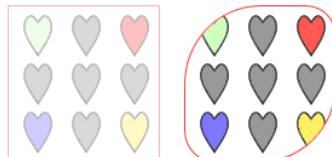
web sayfası görüntüsü

- a) body {background-image:url(border-image.png); background-repeat:no-repeat; background-position:300px -70px;}
- b) img {background-image:url(border-image.png); background-repeat: no-repeat; background-position:300px -70px; }
- c) border-radius {background-image:url(border-image.png); background-repeat: no-repeat; background-position:300px -70px; }
- d) opacity {background-image:url(border-image.png); background-repeat: no-repeat; background-position:300px -70px; }
- e) body {background-image:url(border-image.png); background-repeat: no-repeat; background-position:-70px 300px; }

7. Aşağıdaki stil tanımları aynı resme ayrı ayrı uygulanıyor. Bu sayfanın görüntüsü hangisi olabilir?

```
<style type="text/css">  
    .bir {opacity:0.3; border:1px solid red;}  
    .iki {border-radius:20% 50%; opacity:0.8; border:1px solid red;}  
</style>
```

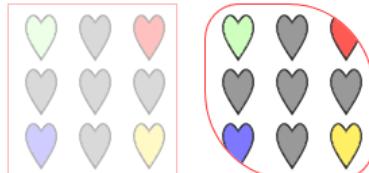
a)



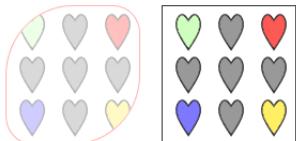
b)



c)



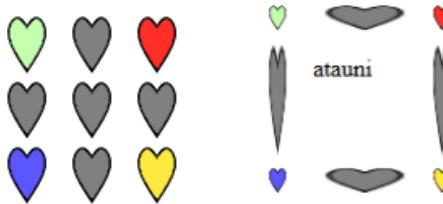
d)



e)



8. Aşağıdaki “border-image.png” dosyası kullanılarak “atauni” metninin web sayfası görüntüsündeki gibi görünmesi sağlanmıştır. Bu işlemin gerçekleşmesini aşağıdaki stil tanımı gerçekleştirebilir?



border-image.png web sayfası görüntüsü

- a) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border: 20px solid transparent; border-image: url(border-image.png) 33.3% round; } </style>`
 - b) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border: 20px solid transparent; background-image: url(border-image.png); } </style>`
 - c) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border-image: url(border-image.png) 33.3% stretch; } </style>`
 - d) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border: 20px solid transparent; border-image: url(border-image.png) 33.3% stretch; } </style>`
 - e) `<style type="text/css"> .cerceve { height:60px; width:60px; padding: 15px; border: 20px solid transparent; border-image: url(border-image.png) 33.3% repeat; } </style>`
9. Arka plan resmi olarak “arka-plan.jpg” dosyasını kullanan, arka plan resmini yatayda tekrarlayan ve arka plan görüntüsünün sayfanın geri kalanıyla birlikte kaymadığı stil tanımı aşağıdakilerden hangisi olabilir?
- a) `body {background-image:url(arka-plan.jpg); background-repeat: repeat-y; background-attachment: fixed;}`
 - b) `body {border-image:url(arka-plan.jpg); background-repeat: repeat-x; background-attachment: fixed;}`
 - c) `body {background-image:url(arka-plan.jpg); background-repeat: repeat-x; background-attachment: fixed;}`
 - d) `img {background-image:url(arka-plan.jpg); background-repeat: repeat-x; background-attachment: scroll;}`
 - e) `body {background-image:url(arka-plan.jpg); background-repeat: repeat-x; background-attachment: scroll;}`

10. Aşağıdaki class stil tanımlarından hangisi, uygulandığı görselin saydamlık değerini %50 ve köşeleri 10 piksel çapında yuvarlamak için uygundur?

- a) `<style type="text/css"> .resim {border-radius:20px; opacity:0.5; </style>`
- b) `<style type="text/css"> img {border-radius:20px; opacity:0.5; border:5px solid green;} </style>`
- c) `<style type="text/css"> img {border-radius:20px; opacity:0.5; </style>`
- d) `<style type="text/css"> .resim {border-radius:20px; opacity:0.5; border:5px solid green;} </style>`
- e) `<style type="text/css"> img {border-radius:20px; opacity:50%; border:5px solid green;} </style>`

Cevap Anahtarı

1.e, 2.a, 3.a, 4.e, 5.b, 6.b, 7.c, 8.d, 9.c, 10.d

YARARLANILAN KAYNAKLAR

- Brown, T. B. (2018). CSS master (2nd Edition). VIC: SitePoint Pty. Ltd.
- Dean, J. (2019). Web programming with HTML5, CSS, and Javascript. MA: Jones & Bartlett Learning, LLC, an Ascend Learning Company.
- Grant, K. J. (2018). CSS in Depth. NY: Manning Publications Co.
- Meyer, E. A. (2018). CSS Pocket Reference (5th Edition). CA: O'Reilly Media, Inc.

JAVASCRIPT İLE KODLAMAYA GİRİŞ



İÇİNDEKİLER

- JavaScript Nereye Yazılır?
- JavaScript Yazma, Çalıştırma ve Test Etme
- JavaScript ile Yapılabilecek İşler
- JavaScript Dilinin Genel Özellikleri



HEDEFLER

- Bu üniteyi çalıştıktan sonra;
 - JavaScript dilini HTML sayfalarına ekleyebilecek,
 - JavaScript yazmak ve çalıştmak için gerekli programları seçebilecek,
 - JavaScript ile yapılabilecek işlemleri planlayabilecek,
 - JavaScript dilinin genel özelliklerini bileyecsiniz.

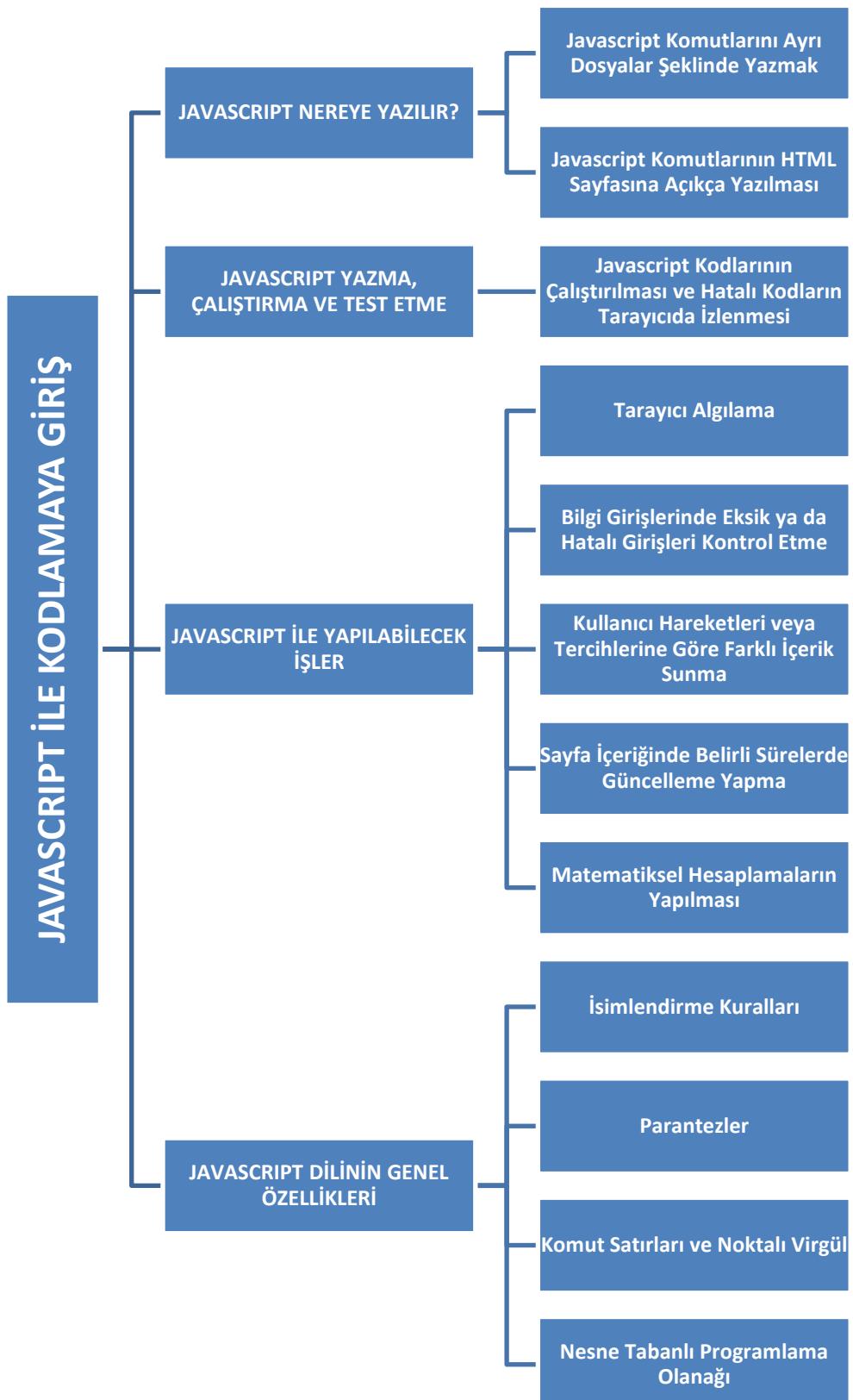


Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET
PROGRAMCILIĞI II
Dr. Öğr. Üyesi Halil
ERSOY

ÜNİTE

9



GİRİŞ

Internetin en büyük görünürlük parçası olan web siteleri bundan yaklaşık 40 yıl önce, 1980'li yılların başında HTML dilinin geliştirilmesiyle ortaya çıkmıştır. Çok büyük miktarda metin türündeki bilgilerin dünya çapında yer alan sunucular üzerinden kullanıcıların bilgisayarlarına aktarılması için geliştirilen HTML dili, ilk ortaya çıktığında çok büyük yenilik getirmiştir. O ana kadar bilgilerin noktadan noktaya özel programlar ile aktarılması söz konusuyken, HTML sayesinde birbirine bağlı biçimde oluşturulan web sayfaları (*hiper metin / hyper text*), web sunucuları üzerinde sürekli barındırılabilir ve istediği anda kullanıcılar tarafından *indirilebilir* olmuştur.



Javascript dili web tarayıcıları tarafından çalıştırılan kodlar için yazılır. Java programlama dili ile ilgisi yoktur.

HTML dili ve bu dil ile geliştirilen web siteleri daha sonraları çok fazla kullanılmaya başlanmıştır. Örneğin sadece sunucuda bulunan bilgilerin kullanıcılarla indirilmesi değil, aynı zamanda kullanıcıların da bu sunuculara bilgi göndermesi için web sayfaları kullanılmaya başlanmıştır. Ayrıca bu hizmetten yararlanmak isteyen kullanıcılar için çok sayıda *tarayıcı* geliştirilmiş (Safari, Internet Explorer, Chrome, Opera, Mozilla vb.) ve farklı cihazların (bilgisayar, tablet, cep telefonu vb.) bu tarayıcıları çalıştırılabilir hâle gelmesi sağlanmıştır. Kullanıcı kitlesi hızla artan ve kullanım platformu hızla çeşitlenen web sayfalarının tüm kullanıcılarla benzer içerik ve hizmet sunabilmesi için, bilginin sunucu ve tarayıcı üzerinde işlenebilmesi amacıyla *HTML diline ek olarak* programlama dilleri geliştirilmiştir.

Web sayfalarının kullanıcıya ulaşmasında *iki farklı aşamada çalışan* programlama dilleri ile veri veya içerik şekillendirilebilir; bunlar *sunucu taraflı* programlama dilleri ve *istemci taraflı* programlama dilleri olarak gruplandırılabilir. *Sunucu taraflı* programlama dilleri, web *sunucuları* üzerinde çalışan, web sitesinin kullanıcı tarafından talep edildiği anda derlenen ve çalışan, ardından sonuç olarak farklı HTML içeriğini önce sunucu üzerinde oluşturup sonra kullanıcıya (istemciye) gönderen dillerdir. En popüler sunucu dilleri arasında PHP, ASP.NET, ASP, CGI sayılabilir. Bu diller ile yazılan programlar, HTML sayfalarının içeriklerinin tamamının ya da bir kısmının her kullanıcı için farklı biçimde olmasını sağlar. Aynı zamanda bu diller ile yazılan sayfalar web sayfaları haricinde kullanıcının görmediği işlemler için de sunucu üzerinde çalışırlar (Örneğin veri tabanı bağlantıları, veri arama ve filtreleme işlemleri, güvenlik işlemleri gibi).



HTML5 standartları öncesinde, Javascript'e alternatif başka diller de mevcuttu, örneğin VBScript.

İstemci taraflı diller ise web sayfasının içinde ya da ek bir dosya olarak kullanıcıya (istemciye) indirilir ve kullanıcının *tarayıcısı tarafından çalıştırılır*. Bu dillerin amacı genellikle kullanıcının web sitesi ile etkileşimi sırasında sayfanın vereceği tepkileri tarayıcı üzerinden çabucak verebilmektir. Örneğin kullanıcının şifresini yazması istenen bir web sayfasında, istemci taraflı bir dil ile bu durum kontrol edilebilir ve şifre girilmeden ilerlenmesi sunucuya bağlantı kurulmaya gerek duyulmadan engellenebilir. En çok kullanılan istemci taraflı dil *Javascript'tir*. Daha önceleri Visual Basic Script (VB Script) ile rekabet içinde olan Javascript, geçtiğimiz yıllarda farklı teknoloji firmaları tarafından tercih edilmiş ve *HTML5 standartları* içerisinde tek başına yer alarak standart hâle gelmiştir.

Bu üitede Javascript diline giriş yapılacaktır. Javascript dil olarak çok daha fazla amaca hitap etse de, bu üitede asıl amaç HTML etiketleri ve CSS kuralları ile oluşturulmuş web sayfalarına daha fazla işlevsellik katmak amacıyla Javascript komutlarının eklenmesidir. Bu nedenle kitabı daha öncedeki ünitelerinde anlatılan HTML ve CSS konularını bildiğiniz varsayılmıştır.

Öte taraftan Javascript diğer web teknolojileri ile birlikte çok farklı amaçlara hizmet edebilecek bir dildir. O nedenle tek bir üitede ele alarak tamamlamak mümkün değildir. Bu nedenle bu kitapta Javascript'e 3 ünite ayrılmış, ancak bunlardan sonra da Javascript ile yazılmış hazır kütüphanelerin kullanımı ile farklı ürünlerin yaratılmasını sağlayacak ünitelere yer verilmiştir.

Bu üitenin kapsamı aşağıdaki gibidir:

- Javascript nereye yazılır?
- Javascript hangi programlar ile yazılır ve çalıştırılır?
- Javascript ile neler yapılabilir?
- Javascript dilinin özellikleri nelerdir?

JAVASCRIPT NEREYE YAZILIR?

HTML sayfalarına Javascript komutlarını yazmanın iki yeri vardır. Bunlardan birincisi komutları “js” uzantılı bir başka metin dosyası içerisinde yazmak ve bu dosyayı HTML sayfasına dâhil etmektir. İkinci yol ise Javascript komutlarını HTML sayfasının içine açık biçimde yazmaktır. Bir HTML sayfasında her iki yöntemi de kullanmak söz konusu olabilir. Bu yolların komutların çalışmasında bir etkisi yoktur; ancak kodu yazan kişinin daha kolay çalışması ve aynı Javascript komutlarının farklı dosyalarda da kullanabilmesi açısından her iki yöntemin avantajları vardır. Aşağıdaki iki başlıkta bu iki yöntem anlatılmıştır.

Javascript Komutlarını Ayrı Dosyalar Şeklinde Yazmak

Aşağıdaki tablonun ilk sütununda HTML ile yazılmış basit bir web sayfasının kodları görülmektedir. Bu kodlardan `<script>` etiketi içerisinde “src” özelliği ile adı belirtilen `scriptKomutlari.js` dosyasının içeriği ise tablodaki ikinci sütunda görülmektedir.

Tablo 9.1. HTML ve Javascript Komutları

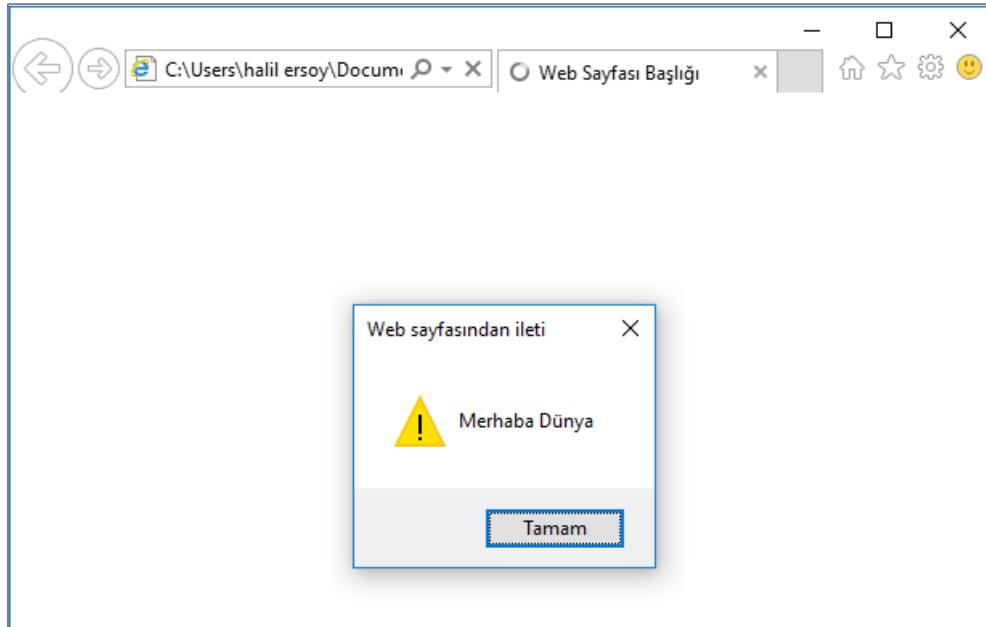
index.html	scriptKomutlari.js
<pre><!DOCTYPE html> <html lang="tr"> <head> <meta charset="UTF-8"> <title>Web Sayfası Başlığı</title> <script src="scriptKomutlari.js"> </script> </head> <body> <h1>Sayfa Başlığı</h1></pre>	<pre>alert("Merhaba Dünya");</pre>

```
<p>  
    Bu bir web sayfasıdır. Sayfadaki ilk paragraf  
    budur.</p>  
</body>
```

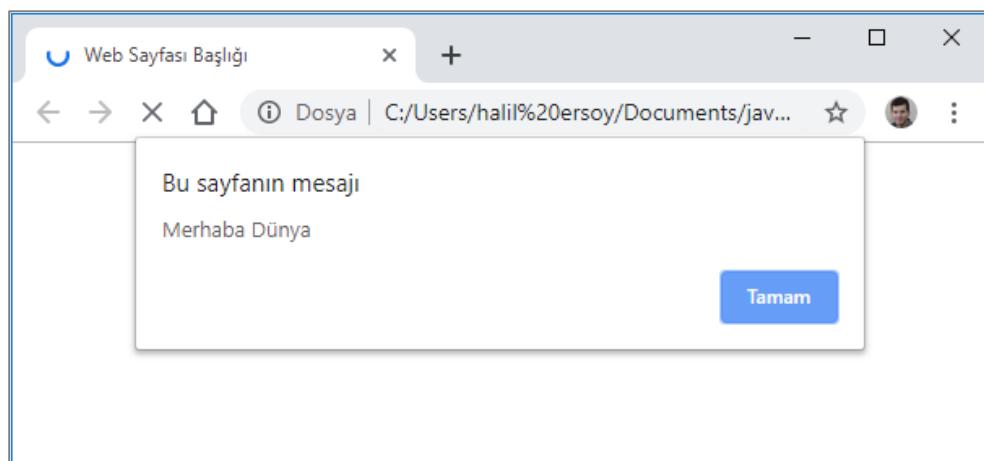
Yukarıdaki “index.html” sayfasının tarayıcıda görünen hâli ise aşağıdaki Şekil 9.1. ve 9.2.’de verilmiştir. Şekil 9.1.’de tarayıcı olarak *Internet Explorer* kullanılırken, Şekil 9.2’de tarayıcı olarak *Chrome* kullanılmıştır.



Tarayıcılar arasında çok az da olsa aynı Javascript komutlarını farklı biçimde çalıştırılabilir.



Şekil 9.1. “index.Html” Sayfasının Internet Explorer ile Görünümü



Şekil 9.2. “index.Html” Sayfasının Chrome ile Görünümü

Öncelikle yukarıdaki iki farklı görüntünün tarayıcı kaynaklı olduğunu belirtmek gerekmek. Farklı firmaların tarayıcıları çok az da olsa aynı Javascript komutlarını farklı biçimde çalıştırır ya da farklı görünümde ekrana aktarır. Ancak bu farklar bu ilk üitede göz ardı edilecektir. Yine de üitede anlatılan komutların doğru öğrenilebilmesi ve test edilebilmesi için tarayıcı olarak bu üitede *Google Chrome’un* tercih edildiğini belirtmek gerekmektedir. Öğrencilerin de birebir aynı çıktıları alabilmeleri için bu tarayıcıyı kullanmaları beklenmektedir.



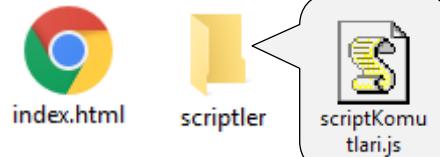
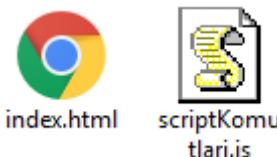
Bireysel Etkinlik

- Bilgisayarınızdaki herhangi bir kelime editörü ile Tablo 9.1.'deki kodları aynı dosyalar hâlinde yapın aynı klasöre kaydedin. Ardından bilgisayarınızdaki farklı tarayıcılar ile HTML sayısını görüntüleyin ve varsa aradaki farklara bakın.



HTML, CSS ve Javascript ile yazılmış web sayfaları mutlaka farklı tarayıcılarında test edilmelidir.

Yukarıdaki örnekte (Tablo 9.1.) gösterilen Javascript yazma tekniğinde, "index.html" dosyası ile "scriptKomutları.js" dosyasının bilgisayarda aynı klasörde yer olması gereklidir. Eğer istenirse, "index.html" dosyasının bulunduğu klasöre "scriptler" adında bir klasör oluşturulabilir ve Javascript dosyası bu klasör altına yerleştirilebilir. Bu yol tercih edilirse, `<script>` etiketinin içindeki "src" özelliği "scriptler/scriptKomutları.js" şeklinde değiştirilmelidir.



```
<script  
src="scriptKomutlari.js">  
</script>
```

```
<script src="scriptler/scriptKomutlari.js">  
</script>
```

Şekil 9.3. Javascript Dosyasının Yerine Göre "Src" Özelliğindeki Fark

Javascript komutlarını bu şekilde HTML sayfalarına "js" uzantılı dosyalar hâlinde çağrıırken aşağıdaki prensipleri bilmek gerekir:

- Bir HTML sayfasına birden fazla Javascript dosyası eklenebilir.
- Bir `<script>` etiketiyle sadece bir Javascript dosyası eklenir. Birden fazla Javascript dosyası eklenmek istenirse, her biri için ayrı ayrı `<script>` etiketi yazılmalıdır.
- Aynı Javascript dosyası bir HTML sayfasına sadece bir kez dâhil edilir.
- Javascript dosyalarının adlarında, tíkla HTML dosya adlarında olduğu gibi noktalama işaretleri yer alamaz. Türkçe karakter ve boşluk karakterleri ise bazı tarayıcılarda çalışsa bile önerilmez.
- Javascript dosyasının adındaki harfler büyük ya da küçük olabilir. Ancak dosya adı ile "src" özelliğindeki dosya adındaki harflerin aynı şekilde olması gereklidir.
- Javascript dosyalarının içinde ayrıca `<script>...</script>` etiketleri yazılmalıdır.
- Javascript dosyaları HTML dosyalarına dâhil edilirken, `<script>` etiketleri genellikle `<HEAD>...</HEAD>` etiketlerinin arasına yazılır. Bu çoğu zaman uygun bir yerdır ancak nadir durumlarda `<HEAD>` etiketleri sonrasında ve `<BODY>...</BODY>` etiketlerinin içinde de yer alabilir.



Javascript dosyaları hem HTML sayfalarına hem de istenirse diğer Javascript dosyaları içine dâhil edilebilirler.

Bu şekilde komutları Javascript dosyalarında tutmanın avantajları şunlardır:

- Aynı Javascript dosyası, başka HTML sayfalarına da dâhil edilebilir, böylece tek bir "js" dosyası tüm HTML sayfaları için gerekli komutları içerebilir.

- Javascript dosyasındaki bir güncelleme, tüm HTML sayfalarına yansıyacağı için değişikliklerin daha hızlı yapılması sağlanır.
- HTML sayfalarına dâhil edilecek Javascript dosyaları farklı bir web sunucusunda yer alabilir. Örneğin aşağıdaki etikette, bir başka web sitesinin adresi ile birlikte verilen JS dosyası HTML sayfasına dâhil edilebilir.
`<script src="http://www.server.com/file.js"></script>`
- HTML dosyasında Javascript komutları yer almayacağı için sayfa daha az koddan oluşur.

Javascript Komutlarının HTML Sayfasına Açıkça Yazılması

HTML sayfalarında etiketler arasında `<script>.....</script>` etiketleri arasında Javascript komutları yazılabılır. Aşağıdaki web sayfasının kodlarını ve devamındaki çıktıyı inceleyelim.

Tablo 9.2. Script Etiketleri Arasına Yazılmış Javascript Komutları

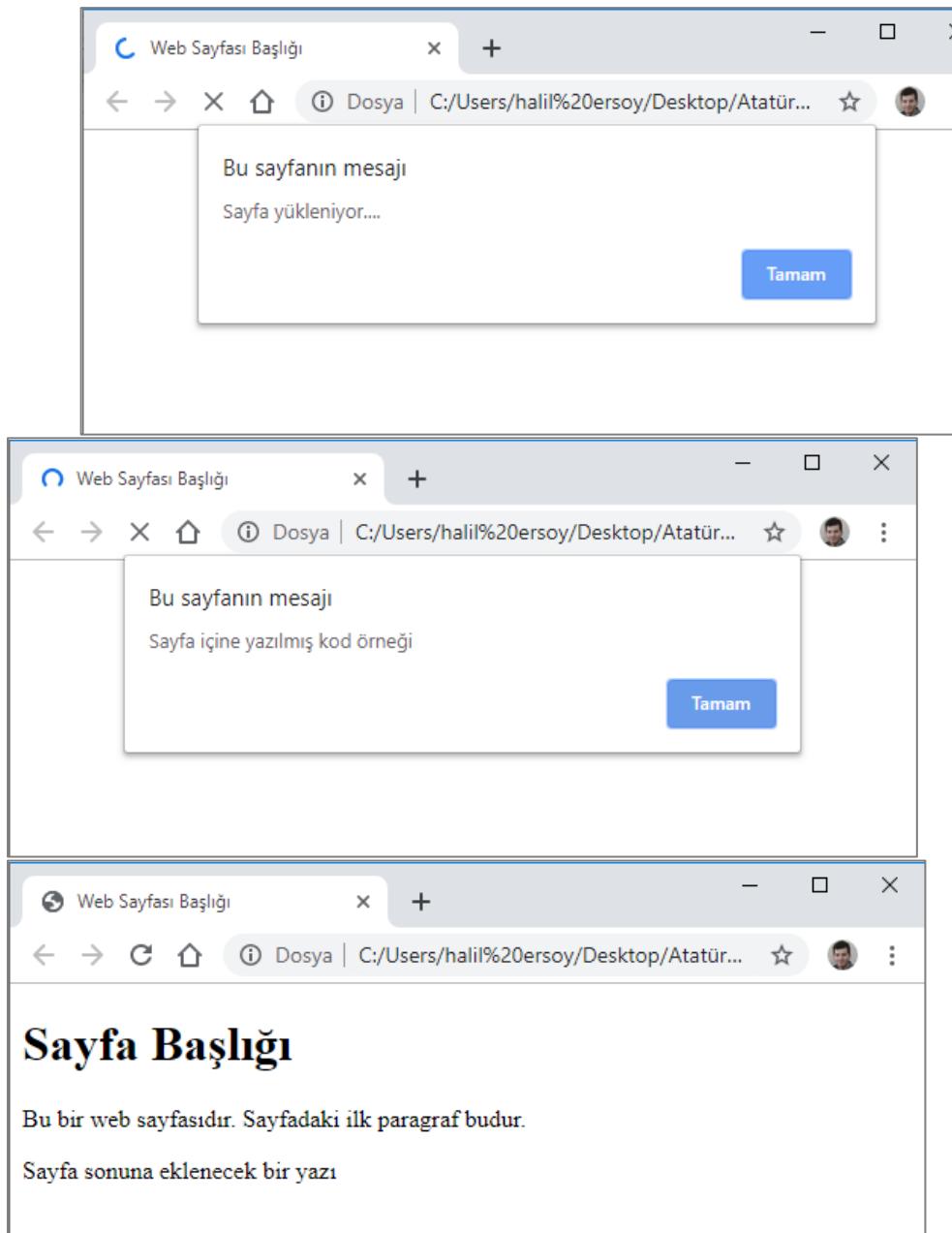
HTML ve Javascript Komutları

```
<!DOCTYPE html>
<html lang="tr">
<head>
    <meta charset="UTF-8">
    <title>Web Sayfası Başlığı</title>
    <script>
        alert('Sayfa yükleniyor....');
    </script>
</head>
<body>
    <h1>Sayfa Başlığı</h1>
    <p>Bu bir web sayfasıdır. Sayfadaki ilk paragraf budur.</p>
    <script>
        alert('Sayfa içine yazılmış kod örneği');
        document.write('Sayfa sonuna eklenecek bir yazı');
    </script>
</body>
```



Bir HTML sayfasında birden fazla yerde Javascript komutları yazılabılır.

Yukarıdaki HTML sayfası tarayıcıya ilk yüklendiğinde, sırası ile aşağıdaki resimlerdeki mesajlar ve içerikler ekrana gelir.



Şekil 9.4. Sayfanın Yüklenmesi Sırasında Ekrana Gelen Mesajlar

Yukarıdaki örnekte görüldüğü üzere, bir HTML sayfasına birden fazla yerde **script etiketleri** (script blokları) ile Javascript komutları yazılabilir. Komutların çalışma sırası yukarıdan aşağıyadır.

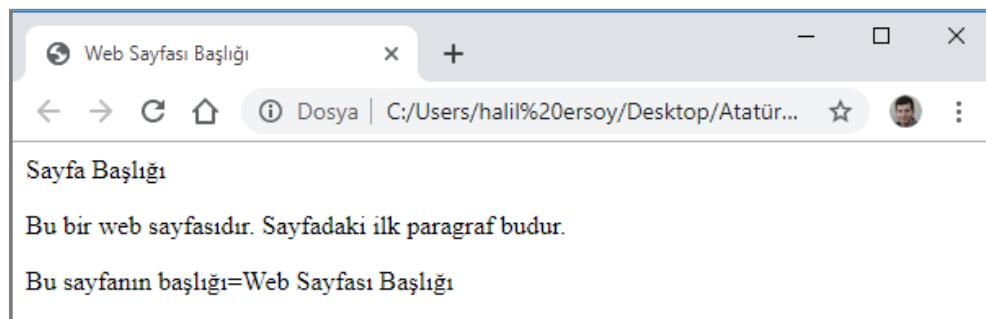
Bu yöntemle yazılan Javascript komutları sadece **yazıldıkları sayfada** çalışırlar. Bir başka HTML sayfasında aynı komutların çalışması istenirse, aynı komutların o sayfaya da yazılması gereklidir.

Javascript komutlarının HTML sayfası içerisinde yazılma yönteminde sıkılıkla kullanılan bir yol ise, komutları **Javascript fonksiyonları** hâlinde yazmak ve bu fonksiyonları istenilen yerden çağrıarak çalıştmaktır.

Tablo 9.3. Javascript Komutlarının Fonksiyon İçinde Yazılması ve Başka Yerden Çağırılması

HTML ve Javascript Komutları

```
<!DOCTYPE html>
<html lang="tr">
<head>
    <meta charset="UTF-8">
    <title>Web Sayfası Başlığı</title>
    <script>
        function sayfaAdresiniGoster(){
            document.write("Bu sayfanın başlığı=" + document.title);
        }
    </script>
</head>
<body>
    <h1>Sayfa Başlığı</h1>
    <p>Bu bir web sayfasıdır. Sayfadaki ilk paragraf budur.</p>
    <script>
        sayfaAdresiniGoster();
    </script>
</body>
```

**Şekil 9.5.** Sayfanın Çıktısı

Javascript komutları
fonksiyonlar hâlinde
yazılabilir.

Tablo 9.3.'teki kodlar ve Şekil 9.5.'teki çıktı incelendiğinde, **<HEAD>** etiketleri arasındaki **script** blokunda, **sayfaAdresiniGoster()** adından bir fonksiyon tanımlanmış görülmektedir. Bu fonksiyon, tanımlanmış olduğu yerde veya sıralamada çalışmaz, ancak daha sonra adının bir komut gibi yazıldığı zaman çalışır. **<BODY>** etiketleri arasındaki **script** blokunda bu fonksiyonun adı yazılarak çalışması sağlanmıştır. Fonksiyonun içindeki **document.write()** komutunun çıktısı olan yazı, fonksiyonun tanımlanmış olduğu yerde değil, **çağrıldığı yerde** ekrana yazılmıştır.

Fonksiyonlar konusu daha sonraki ünitelerde detaylı biçimde ele alınacaktır. Ancak bu aşamada kodları yazma yeri olarak açıklanmak istenmiştir. Fonksiyon tanımlamada kullanılacak fonksiyon ismi, fonksiyonun türü, alacağı parametreler gibi detaylar bundan sonraki ünitelerde “Fonksiyonlar” başlığı ile ele alınacaktır.



Bireysel Etkinlik

- Şekil 9.8.'deki kodları editörde yazıp sayfayı tarayıcınızda görüntüleyin.
- Sayfadaki <title> etiketindeki metni değiştirip sayfanızı kaydedin ve tarayınızı yenileyin. Aradaki farkı görün.

JAVASCRIPT YAZMA, ÇALIŞTIRMA VE TEST ETME



Javascript komutları metin tabanlı yazılır ve tarayıcı üzerinde derlenip çalıştırılır.

Javascript komutları tipki HTML etiketleri gibi metin tabanlı yazılır ve tarayıcı üzerinde derlenip çalıştırılır. Bu nedenle HTML komutlarını hangi program ya da editörde yazıyorsanız, aynı program veya editörde Javascript komutlarını da yazabilirsiniz. Hatta HTML ve Javascript'i basit metin editörlerinde bile (*Not Defteri®* gibi) yazabilirsiniz.

Öte taraftan web sayfası geliştirmek ve/veya tasarlamak çoğu zaman çok fazla sayıda kod yazmayı ve sonucu görmeyi gerektirir. Bu aşamada programcılara yardımcı olacak birçok profesyonel editör kullanılmaktadır. Bu editörler sadece HTML etiketleri değil, aynı zamanda CSS ve Javascript komutlarını yazma aşamasında da çok fazla yardımcı olmaktadır.



Örnek

- Hem HTML sayfalarını hem de Javascript komutlarını yazmak için aşağıdaki editörler kullanılabilir:
 - Adobe Dreamweaver® (Ücretli)
 - Microsoft Visual Studio® (Ücretli)
 - Brackets® (Ücretsiz)
 - Notepad++® (Ücretsiz)
 - Visual Studio Code® (Ücretsiz)



Javascript dilinde yapılan yazım yanlışları tarayıcılar tarafından algılanabilir. Ancak olası mantık hataları tarayıcı tarafından algılanmayabilir.

Örnekteki editörler ilk akla gelenlerdir ve her geçen gün yeni editörler geliştirilmektedir. Bu üitede anlatılan Javascript komutları için ücretsiz olan *Visual Studio Code®* kullanılmıştır.

Javascript Kodlarının Çalıştırılması ve Hatalı Kodların Tarayıcıda İzlenmesi

Yukarıdaki editörler HTML sayfalarını etiketler, CSS kuralları ve Javascript komutları ile yazma aşamasında yardımcı olurken, yazılan kodların çıktısını göstermemekte veya eksik göstermektedir.

Bu nedenle hangi editörü kullanıyor olursanız olun, HTML sayfalarının ve içerisinde Javascript komutlarının çıktısını görmek için gerçek tarayıcıları kullanmanız tavsiye edilir.

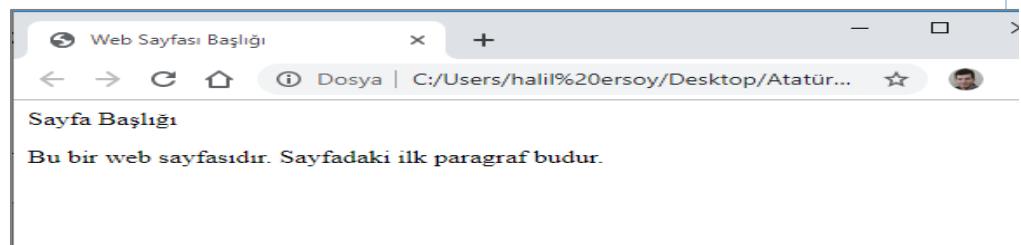
Günümüzde popüler olan tüm tarayıcılar (Google Chrome®, Microsoft Edge®, Microsoft Internet Explorer®, Safari®, Opera®, Firefox® vb.) Javascript'i derleyip çalıştırabilmektedir. Hatta Javascript komutlarında olası bir hatayı kendi içerisinde ayrı bir pencerede gösterebilmektedir. Bu sayede programcılar yazdıkları kodun doğru çalışmadığı zamanlarda hatanın yerini ve olası çözümleri kolaylıkla bulabilmektedir.

Bu durumu göstermek için aşağıdaki kodlarda bilerek **hata** yapılmıştır. Script bloku içinde hatalı biçimde document.vrite() komutu kullanılmıştır. Komutun doğrusu document.write() olmalıdır. Ancak hatalı biçimde yazılan bu kod ile sayfa tarayıcıda açıldığında Şekil 9.6.'daki çıktı ekrana gelecektir.

Tablo 9.4. Hatalı Javascript Komutu İçeren Sayfa Kodları

HTML ve Javascript Komutları

```
<!DOCTYPE html>
<html lang="tr">
<head>
    <meta charset="UTF-8">
    <title>Web Sayfası Başlığı</title>
    <script>
        function sayfaAdresiniGoster(){
            document.vrite("Bu sayfanın başlığı=" + document.title);
        }
    </script>
</head>
<body>
    <h1>Sayfa Başlığı</h1>
    <p>Bu bir web sayfasıdır. Sayfadaki ilk paragraf budur.</p>
    <script>
        sayfaAdresiniGoster();
    </script>
</body>
```



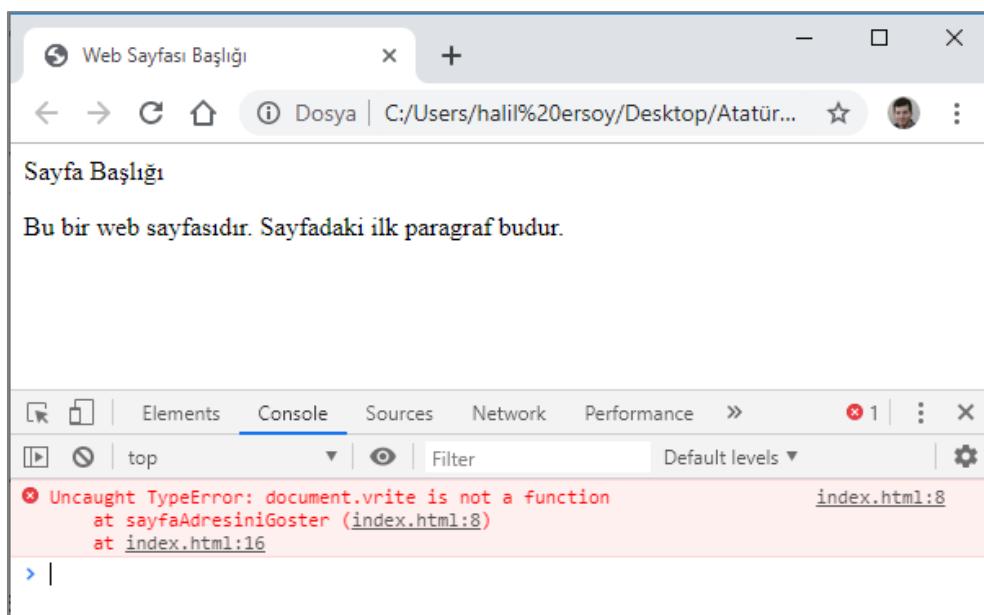
Şekil 9.6. Hatalı Javascript Komutunun Olduğu Sayfanın Çıktısı



Javascript kodlarındaki yazım (sözdizim) hataları HTML etiketlerinin çalışmasını engellemez.

Javascript kodlarındaki *yazım (sözdizim) hataları* HTML etiketlerinin çalışmasını engelmez. Ancak Javascript komutlarının çalışması, hatalı yerden itibaren durur ve devamındaki Javascript komutları çalıştırılmaz. Örnekteki çıktıda Javascript komutları ile yazılması gereken yazının ekrana gelmediğini fark edebilirsiniz.

Hatanın hangi satır ve neden olduğunu anlamak için tarayıcıların “*geliştirici seçenekleri*” (*developer options*) açılmalıdır. Örnekte kullanılan *Google Chrome®* için bu seçeneği klavyeden F12 tuşuna basarak açabilirsiniz. Bu seçeneği açtığınızda tarayıcınızın alt kısmında sayfanın kodlarıyla ilgili bilgiler alabileceğiniz pencereler açılır (Şekil 9.7.).



Şekil 9.7. Tarayıcıda Açılan Geliştirici Seçenekleri

Tarayıcıda açılan *geliştirici seçenekleri* penceresinde *Console* sekmesini seçtiğinizde, varsa *Javascript hatalarını* ve hatanın tespit edildiği *satır numarasını* görebilirsiniz. Örnekteki uyarıda index.html sayfasının 8. satırında tanımlanamaya komut (`document.vrite`) olduğu mesajı verilmektedir. Bu hata mesajını gören programcı, index.html dosyasındaki 8. satıra dönüp hatalı komutu düzeltmelidir.

JAVASCRIPT İLE YAPILABİLECEK İŞLER



Javascript ile içinde bulunduğuğunuz bir sayfadaki herhangi bir form alanına girilen eksik bilgi tespit edilebilir.

HTML sayfalarında Javascript ile yapabileceğimiz sayısız iş vardır. Javascript'in dil kurallarına geçmeden önce bu işlerden en çok popüler olanları kısaca aşağıdaki gibi sıralayabiliriz. Aşağıda anlatılan işlerin Javascript ile nasıl yapıldığı ise daha sonraki ünitelerde anlatılacaktır.

Tarayıcı Algılama

Web sayfalarını ziyaret eden kullanıcılar çok farklı tarayıcı ve hatta cihaz ile bu ziyaretlerini yapmaktadır. Farklı cihaz ve tarayıcılar kimi web sitelerini farklı biçimde algılamakta, yazılmış olan kimi CSS ve Javascript komutlarını farklı biçimde çalıştırmaktadır. Farklılık web sitesi geliştiriciler için can sıkıcıdır çünkü aynı web

sayfasının farklı tarayıcılar için farklı versiyonlarının yazılması gerekmektedir. Farklı yazılmış web sayfalarının hangi kullanıcıya hizmet edeceğini ise kullanıcının tarayıcısının adının, versiyonun ve tarayıcının çalıştığı cihaz hakkında ön bilgi edinilmesini gerektirir.

Bir HTML sayfasının başında yazılacak Javascript komutları ile tarayıcının adı ve farklı özellikleri öğrenilebilir; ardından kullanıcıya kendi tarayıcısı ile uyumlu içerik veya sayfalar gösterilebilir.

Bilgi Girişlerinde Eksik ya da Hatalı Girişleri Kontrol Etme

Günümüzde birçok web sitesi, sadece bilgi sunmak değil aynı zamanda kullanıcıdan bilgi alarak farklı hizmetler sunmaktadır. Kullanıcının girmesi gereken bilgilerin eksik ya da hatalı olması, web sitesinin hizmeti verememesine ve daha önemlisi olası bir hatadan dolayı web sunucusunun sıkıntı yaşamamasına neden olabilir. Örneğin web sayfalarındaki formlarda kullanıcıların girmesi gereken kimi bilgilerin sunucuya gönderilmeden önce biçim olarak kontrol edilmesi ve olası eksikliklerde bu bilginin sunucuya hiç gönderilmemesi gereklidir.

Javascript ile içinde bulunduğu sayfadaki herhangi bir form alanına girilen eksik bilgi tespit edilebilir ve kullanıcıya derhâl uyarı mesajı verilebilir. Örneğin e-posta adresinin girilmesi gereken bir kutuda "@" simbolünün olmaması Javascript kodları ile denetlenip, kullanıcıya "Yanlış ya da eksik bir e-posta adresi girdiniz" uyarısı verilebilir.

Kullanıcı Hareketleri veya Tercihlerine Göre Farklı İçerik Sunma



Yapılabilecek bu işlerde Javascript haricinde yollar da kullanılabilir, ancak Javascript hem hızlı hem de sunucuya yormayan bir çözümüdür.

Çoğu web sitesi, kullanıcıların sayfalar içerisinde hareketlerini (tıklamalar, fare ile üzerinden geçme, listeden bir seçeneğin seçilmesi, takvimde belirli bir gün seçilmesi vb.) algılamak ve bu harekete cevap olarak farklı içerikler göstermek ister. Örneğin seyahat biletini arayan bir yolcunun hareket noktasını seçikten sonra olası varış noktalarının listelenmesi kullanıcı için büyük kolaylık olacaktır.

Javascript ile kullanıcıların farklı içerik öğeleri ile etkileşimi (tıklama, seçme, iptal etme vb.) algılanabilir ve sonrasında uygun içerik, sayfaya yansıtılabilir.

Sayfa İçeriğinde Belirli Sürelerde Güncelleme Yapma

Anlık değişimlerin fazla olduğu konularda, web siteleri, kullanıcılarına sürekli yeni bilgiyi sunmak isterler. Örneğin döviz kurlarını içerik olarak sunan web sitesi, kullanıcı web sayfasına bağlandığı andan itibaren her 5 saniyede bir döviz kurlarını güncel hâliyle göstermek isteyebilir.

Kullanıcının herhangi bir hareketine bağlı olmayan bu güncelleme işlemi, Javascript ile hazırlanmış ve belirli periyotlarda tekrar çalışan fonksiyonlar ile yapılabilir.

Matematiksel Hesaplamaların Yapılması

Kullanıcılar kendi girecekleri bazı rakamlar ile önceden yazılmış formüller sayesinde hesaplamalar yapmak isteyebilirler. Örneğin üniversite tercihlerinde

kullanılacak bir web sayfasında, kullanıcının sayfaya gireceği doğru ve yanlış cevap sayıları ile netleri hesaplanabilir ve bu netlere göre puan hesaplanabilir.

Javascript ile kullanıcıların sayısal veya sözel veri girmesi ve bu verilerin belirli formüllerde hesaplanarak sonuçların ekran yazılması sağlanabilir.

JAVASCRIPT DİLİNİN GENEL ÖZELLİKLERİ

Javascript diğer programlama dilleri gibi kendisine özgü özellikleri olan bir dildir. Dili oluşturan farklı öğelerin sözdizimi ve özellikleri bu ve ilerleyen ünitelerde ele alınacaktır. Ancak Javascript dilinin genel özellikleri aşağıdaki alt başlıklarda belirtilmiştir.

İsimlendirme Kuralları

Javascript'te programcı tarafından yaratılacak bir değişken, fonksiyon ya da nesnenin ismi aşağıdaki kurallara uymak zorundadır. Aksi takdirde kodlar hata verecektir.



En çok yapılan kodlama hatlarının başında, bir değişken ya da fonksiyonun tanımlandıktan sonra farklı şekilde (büyük/küçük harf) kullanılmaya çalışılmasıdır.

- Değişkenler, fonksiyon isimlendirmeleri ve komutlarda *büyük ve küçük harf ayımı vardır*. Örneğin alert() komutu doğru, Alert() ifadesi ise geçersizdir.
- Değişken, fonksiyon ve diğer isimlendirmelerde İngiliz alfabetesindeki *harfler* (A-Z / a-z), *rakamlar* ve *alt tire* (_) ve *dolar* (\$) sembollerini kullanılabılır. Bunların dışında sembol kullanılmasına izin verilmez.
- Değişken, fonksiyon ve diğer isimlendirmelerde *ilk karakter rakam olamaz*, mutlaka harf veya alt tire karakteri olmalıdır.
- \$ simbolünün *ilk karakter* olarak kullanılması önerilmez, bu Javascript kütüphaneleri ile birlikte kullanıldığında çakışmalara neden olabilir.
- Javascript *diline has komut ve ifadeler*, isimlendirmede kullanılamazlar.
- Kullanılan isimdeki karakter sayısı *sınırsızdır*.
- Aynı *ismide* iki değişken, fonksiyon ya da nesne aynı web sayfası tarafından aynı anda tanımlanamaz. Ancak fonksiyonların kendi içindeki lokal değişkenler, bir başka fonksiyon içinde yine aynı isimle lokal olarak tanımlanabilirler.

Aşağıdaki tabloda geçerli ve geçersiz isimlendirme örnekleri mevcuttur.

Tablo 9.5. Geçerli ve Geçersiz Isimlendirme Örnekleri

Geçerli isimlendirmeler	Geçersiz isimlendirmeler
• isim	• dosya ismi
• ogrenci	• öğrenci
• Nokta1	• 1Nokta
• s1	• s-1
• ses_seviyesi	• ses.seviyesi
• GENISLIK	• GENISLIK?
• A255xl	• 255Axl
• _hataKodu	• _hata%Kodu
	• 5000

- function
- if

Parantezler

Javascript'teki bazı komutların yapısı gereği, ilgili bazı ifadelerin veya komutların gruplanması gerekebilir. Bu grupları oluşturmak için **normal parantezler** () ya da **küme parantezleri** { } kullanılır. Hangi komutta hangi parantezin kullanılacağı ileride anlatılacaktır, ancak önemli olan açılmış olan bir **parantezin mutlaka kapatılması** gereklidir. Aşağıdaki örnekteki her türlü parantezin mutlaka kapatılmış olduğunu görebilirsiniz.

Tablo 9.6. Normal Ve Küme Parantezleri Açıldıktan Sonra Kapatılmalıdır.



Yazılan kodlarda düzgün hizalama ve isimlendirme, olası hataların yapılmasını engeller.

Javascript Komutları

```
<script>
var birSayi=11;
if((birSayi % 2) == 0)
{
    alert(birSayi+" sayısı çift sayıdır.");
}
else
{
    alert(birSayi+" sayısı tek sayıdır.");
}
</script>
```

Eğer açılan bir parantez (normal ya da küme parantezi) daha sonra kapatılmaz ise bu **yazım hatası** olarak karşımıza çıkar ve kodlar o noktadan sonra çalıştırılmaz.

Komut Satırları ve Noktalı Virgül

Popüler birçok dile benzer biçimde Javascript'te komutlar **noktalı virgül** karakteri ile biter, ardından bir başka komut başlar. Bu özellik farklı komut ve ifadelerde detaylıca gösterilecektir, ancak aşağıdaki örnekte, her satırın sonunda olan noktalı virgül (;) karakterinin, o komutu bitiren sembol olduğu anlamına geldiği fark edilmelidir. Bu sembol sayesinde bir komutun bittiği anlaşılacağı için, eğer istenirse komutlar bu sembol ile birlikte **yan yan** da yazılabilir. Aşağıdaki iki komut bloku aynı şekilde geçerlidir.

Tablo 9.7. Komut Sonlarındaki Noktalı Virgül Karakteri

Javascript Komutları

```
<script>
var a=5;
var b=4;
document.write("toplam= " + (a + b));
</script>
```

```
<script>
var a=5; var b=4; document.write("toplam= " + (a + b));
</script>
```

Benzer biçimde bir komut noktalı virgül ile bittikten sonra, bir sonraki komut başlamadan önce, araya istenildiği kadar **boşluk** bırakılabilir. Aşağıdaki komut bloku bu anlamda geçerlidir.

Tablo 9.7. Gereksiz Boşlukların Yer Aldığı Komutlar

Javascript Komutları

```
<script>
var a=5;  var b=4;

document.write("toplam= " + (a + b));
</script>
```

Öte taraftan yukarıdaki 3 alternatiften *ilkini* kullanmanız önerilir, çünkü komutların programcı tarafından *anlaşılır* biçimde yazılması profesyonel hayatı çok önemlidir. Yazılan komutların aradan zaman geçtikten sonra tekrar gözden geçirilmesi gerektiğinde ya da farklı bir programcı tarafından düzenlenmesi gerektiğinde, hizalamalar ve düzgün yazılmış kodlar anlamayı kolaylaştıracaktır.

Nesne Tabanlı Programlama Olanlığı



Nesne tabanlı programlama yapabilmek için bu konuda tek başına bir ders alınması tavsiye edilir.

Javascript, nesne tabanlı programlama dilidir. Hem yeni sınıflar ve nesneler yaratmaya, hem de HTML etiketleri ile oluşturulmuş HTML belgesinin nesnelerine ulaşarak kodlama yapmayı sağlar.

Öte taraftan bu kitaptaki Javascript diline ayrılmış 3 üitede, nesne tabanlı programlama kapsamında sınıf ve nesne oluşturma ele alınmamıştır. Ancak `document.title.ToString()` komutunda olduğu gibi, Javascript ile var olan HTML sayfasındaki nesnelerin kullanımı anlatılacaktır.

Nesne tabanlı programlama, yazılan kodların tasarımını kolaylaştırır. Aynı zamanda kodların farklı programcılar tarafından bireysel ihtiyaçlar doğrultusunda tekrar tekrar kullanılmasını sağlar. Son olarak kodların güncellenmesinin kolaylaşmasını ve gereksiz detayların gizlenerek daha hızlı algoritma oluşturulmasını sağlar.



:Özet

- HTML sayfalarının içeriği, sayfayı ziyaret eden kişinin talepleri, kullanıcının tarayıcı veya cihazı ile bağlantılı olarak değişik biçimlerde görüntülenebilir. Bunu saptamak üzere HTML içeriklerine müdahale edebilen programlama dilleri geliştirilmiştir. Bu diller sunucu taraflı ve istemci taraflı programlama dilleri olarak ikiye ayrılabilir. Sunucu taraflı diller HTML içeriklerini sunucu üzerinde düzenleyerek sayfa yüklenmeden önce sunucu üzerinde HTML sayfalarını hazırlar. Ardından bu kodlar kullanıcının web tarayıcısına (istemci) gönderilir. Tarayıcıya ulaşan (indirilen) kodlar, web sayfası olarak gösterilmek üzere son kez derlenir. Bu derlenme aşamasında HTML, Javascript ve CSS kodları çalıştırılarak kullanıcının ekranına web sayfası oluşturulur.
- Javascript, istemci taraflı bir dildir. Tarayıcılar tarafından çalıştırılır. HTML etiketleri ile oluşturulmuş web sayfasının içeriğini veya görünümünü ekranda görüntülenmeden önce veya sonra değiştirebilir.
- Javascript Nereye Yazılır?**
- Javascript kodları HTML sayfaları içerisinde iki yolla yazılabilir.
- Bunlardan ilki HTML sayfası dışında ayrı bir Javascript kod dosyası oluşturup (örneğin kodlar.js) bu dosyayı HTML sayfası içerisinde çağrılmaktır. Bu yöntemde çağrıma genellikle HTML sayfasındaki <HEAD>...</HEAD> etiketleri arasında yapılır. Ancak bu zorunlu değildir, sayfadaki başka bir noktada da çağrı yapılabilir. Çağrı komutu HTML etiketlerinden biri olan <script>...</script> ile yapılır. Örneğin <script src="kodlar.js"></script> komutudur, içinde bulunduğu HTML sayfası ile aynı klasörde bulunan "kodlar.js" dosyasını HTML sayfasına dahil eder.
- İkinci yöntem ise Javascript komutlarını HTML sayfaları içerisine açıkça yazmaktır. Bu yöntemde yine <script>...</script> etiketleri kullanılır. Bu defa bu etiketlerin arasına Javascript komutları açıkça yazılır. Örneğin <script>document.write("Merhaba dünya");</script> satırı, HTML sayfasında nerede kullanılırsa oraya "Merhaba Dünya" ifadesini yazar.
- Javascript Yazma, Çalıştırma ve Test Etme**
- Javascript tipki HTML sayfaları gibi metin tabanlı bir editörde yazılabilir. Çoğu geliştirici HTML sayfalarını kodlamak üzere kullandığı web editörü ile Javascript kodlamaktadır.
- Öte taraftan herhangi bir metin editöründe (Not Defteri® gibi) bu dosyalar açılıp yazılabilir.
- Javascript kodlarını çalıştırmak için kodları dahil ettiğiniz HTML sayfasını tarayıcınızda açmalısınız. Bu aşamada tavsiyemiz, popüler olan tüm tarayıcılarda sayfanızı açmanızdır. Bunun sebebi, Javascript'in çok nadir de olsa farklı tarayıcılarda farklı şekilde çalıştırılmasıdır. Hatta günümüzde cep telefonu ve tablet kullanımının artmasıyla denemelerinize bu cihazları da eklemenizi tavsiye ederiz.
- Yazılan Javascript komutlarında yazım (sözdizim) hataları veya mantık hataları olabilir.Çoğu tarayıcı yazım hatalarını algılar ve hatanın olduğu satırдан itibaren Javascript komutlarını çalıştmayı durdurur. Bu durumda web sayfasının HTML komutları yine de çalışır.Çoğu kullanıcı bunun farkına varmaz, ancak sayfadaki beklenmeyen içerik veya hareketleri fark edebilir. Geliştirici olarak yapmamız gereken tarayıcılardaki "geliştirici seçeneklerini" aktif hale getirip, Javascript hatalarını görmek ve editörümüze dönüp düzeltmeler yapmaktadır.
- Eğer kullandığınız editör Javascript diline destek veriyorsa, kod yazma aşamasında yanlış yazımları fark ederek uyarı verir. Tavsiyemiz bu özellikle bir editör kullanmanızdır.



Özet(devamı)

- Kodlarda yapılan mantık hataları ise web sayfasında herhangi bir hata uyarısı neden olmaz ve geliştirici seçenekleri ile de ortaya çıkmaz. Bu hatalar çıktıının farklı olması ya da kodların bir süre sonra çalışmaması ile ortaya çıkar. Bu durumda geliştirici tekra rkodelarına dönüp düzeltmeler yapmalıdır. Mantık hatalarından korunmanın en iyi yolu yazılan kodları farklı girdiler veya farklı ortamlarda çalıştırarak olası aksilikleri önleyecek kodları ilave etmektedir. Bu aşamada tarayıcıların "geliştirici seçenekleri" yine kodların çalışmasını izlemek için (hata olmasa bile) kullanılabilir.
- **Javascript ile Yapılabilen İşler**
- Javascript ile yapılabilecek işler çok fazladır. Bu kitapta bunlardan en çok tercih edilenleri anlatmaya çalışılmıştır. Detayları daha sonra anlatılacak olsa da geliştiricilere fikir vermek amacıyla aşağıdaki işlerin Javascript ile yapılabilecek işler olduğu bilinmelidir.
- Tarayıcı algılama, web sitesi sahiplerinin farklı tarayıcı kullanan kullanıcılarla hizmet sunmasında son derce gerekli bir işlemidir. Farklı tarayıcılara göre farklı içerik, CSS veya Javascript komutları gerekebilir. Bunu algılamak için yine Javascript komutları kullanılır.
- Bilgi girişlerinde eksik ya da hatalı girişleri kontrol etme, bir sonraki adımda hata olabilecek durumları engellemek için yapılması gereken işlemlerdir. Örneğin eposta adresi girilmesi gereken bir kutunun boş bırakılması ya da eposta adresi olmayan bir ifadenin girişmiş olması Javascript ile kontrol edilebilir ve tespit edildiğinde daha fazla işlem yapmadan kullanıcı uyarılabilir.
- Kullanıcı hareketleri veya tercihlerine göre farklı içerik sunma, sayfalarımıza etkileşim katmanın bir yoludur. Örneğin kullanıcının bir resim ya da metin üzerine fare ile gelmesi sonucunda o resim ya da metnin daha büyük görünmesi gibi etkileşimler Javascript ile yapılabilir.
- Sayfa içeriğinde belirli sürelerde güncelleme yapma günümüzde çok fazla yapılmaktadır. Anlık değişimleri çok olduğu web sitelerinde, her 10 saniyede ya da her 10 dakikada bir içeriğin belirli bir kısmının (tamamı da olabilir) yenilenmesi gerekebilir. Belirli periyodlarda yapılacak bu güncelleme Javascript ve sunucu taraflı dillerin işbirliği ile yapılabilir.
- Matematiksel hesaplamaların tarayıcı ekranında yapılması, kullanıcılar için basit hesaplamalarda farklı yazılımlara gerek duymadan sonucu hesaplamasını sağlar. Javascript ile hesap makinesi gibi oluşturulmuş ekranlar üzerinden işlem yapılabilir.
- **Javascript Dilinin Genel Özellikleri**
- Genel sözdizim özellikleri açısından Javascript C/C++ diline benzer. Bu kapsamında isimlendirme, komut yazım stili ve nesne tabanlı programlama desteği açısından bu dillere benzerlik gösterir.

DEĞERLENDİRME SORULARI

1. Aşağıdakilerden hangisi istemci taraflı bir dildir?
 - a) PHP
 - b) ASP.NET
 - c) CSS
 - d) CGI
 - e) ASP
2. Javascript dili ile yazılmış kodlar aşağıdaki programların hangisi ile çalıştırılabilir?
 - a) Not Defteri®
 - b) Adobe Dreamweaver®
 - c) Visual Studio Code®
 - d) Internet Explorer®
 - e) NotePad++®
3. Aşağıdakilerden hangisi bir Javascript dosyasını HTML sayfasının içerisinde doğru şekilde dâhil eden komuttur?
 - a) <script "kodlar.js"></script>
 - b) <script file="kodlar.js"></script>
 - c) <script>"kodlar.js"</script>
 - d) <script>kodlar.js</script>
 - e) <script src="kodlar.js"></script>
4. Aşağıdakilerden hangisi bir Javascript dosyasını HTML sayfasına dâhil ederek kullanmanın avantajlarından biri değildir?
 - a) Birden fazla HTML sayfasında aynı komutları kullanabilmek
 - b) Komutları sunucu tarafında çalıştırabilmek
 - c) Komutları düzeltmede zaman kazanmak
 - d) Başka sunucular üzerindeki dosyaları dâhil edebilmek
 - e) HTML kodlarında satır sayısını azaltmak
5. Javascript komutları HTML kodları içerisinde nereye yazılmalıdır?
 - a) HEAD etiketleri arasına
 - b) BODY etiketleri arasına
 - c) P etiketleri arasına
 - d) HTML etiketleri başlamadan önce
 - e) SCRIPT etiketleri arasına

6. Javascript komutlarındaki yazım hatalarını tespit etmek için tarayıcıların hangi özelliğini aktif hâle getirmek gereklidir?
 - a) CSS desteği
 - b) Tarayıcı geçmişi
 - c) Sık kullanılanlar özelliği
 - d) Geliştirici seçenekleri
 - e) Tam ekran desteği
7. Javascript komutlarında eğer bir yazım hatası bulunursa, sayfa tarayıcıda çalıştırılmak istendiğinde aşağıdakilerden hangisi gerçekleşir?
 - a) Hatalı ve sonraki Javascript komutları çalıştırılmaz.
 - b) Web sayfası görüntülenmez.
 - c) Tarayıcı Javascript komutlarının tümünü çalıştırılmaz.
 - d) Tarayıcı ekranına Javascript hatası ile ilgili uyarı mesajı gelir.
 - e) Tarayıcı hatalı kodu düzeltici adım atabiliyorsa düzeltir ve çalıştırır.
8. Aşağıdakilerden hangisi Javascript ile yapılabilecek işlerden biri değildir?
 - a) Matematiksel işlem yapmak
 - b) İstenmeyen tarayıcı algılandığında bilgisayarda farklı tarayıcı açmak
 - c) Her dakika ekranındaki hava durumu bilgisini güncellemek
 - d) Eksik girilen telefon numarasını tespit etmek
 - e) Bir resmi, tıklandığında büyütmek
9. Aşağıdakilerden hangisi Javascript'te tanımlanabilecek geçerli bir isimdir?
 - a) RESIM BOYU
 - b) Resim-boyu
 - c) resimBoyu
 - d) 40Resim Boyu
 - e) !ResimBoyu
10. Javascript komutlarının sonuna hangi karakter konur?
 - a) ;
 - b) .
 - c) !
 - d) /
 - e) \

Cevap Anahtarı

1.c, 2.d, 3.e, 4.b, 5.e, 6.d, 7.a, 8.b, 9.c, 10.a

YARARLANILAN KAYNAKLAR

Javascript Tutorial. 6 Temmuz 2019 tarihinde <https://www.w3schools.com/js/> adresinden erişildi.

Javascript Dersleri. 7 Temmuz 2019 tarihinde <http://javascript.sitesi.web.tr/> adresinden erişildi.

JAVASCRIPT DİLİ VE KOMUTLARI



İÇİNDEKİLER

- Javascript ve HTML Etiketleri İlişkisi
- Değişkenler
- Operatörler
- Koşul Komutları ve Koşul Operatörü



HEDEFLER

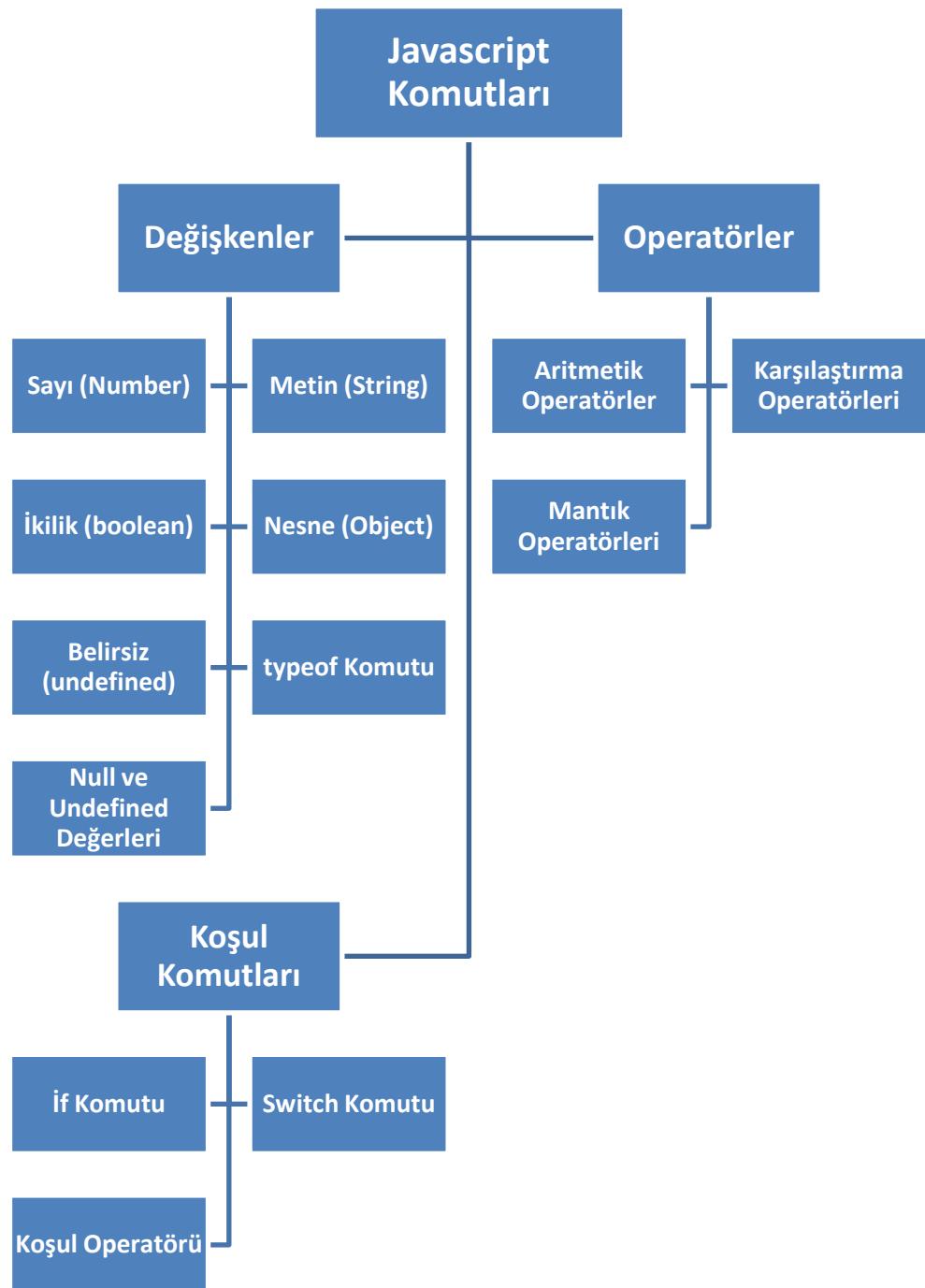
- Bu üniteyi çalıştıktan sonra;
- Javascript kodları ile HTML etiketlerine ve sayfalarına çıktı yazabilecek,
- Değişken tanımlayabilecek ve uygun türü seçebilecek,
- Değişken ve/veya değerler ile işlem yapmak için gerekli operatörleri kullanabilecek,
- Bir koşula bağlı olarak programın akışını yönlendirebileceksiniz.



Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET
PROGRAMCILIĞI II
Dr. Öğr. Üyesi Halil
ERSOY

ÜNİTE
10



GİRİŞ

Javascript, dil özellikleri bakımında C/C++ programlama dillerine benzer. Bu benzerliğin ilk akla gelen örneği komutların noktalı virgül ile sonlanması gerektiği ve büyük/küçük harf ayrimının var olmasıdır. Öte taraftan komutların gruplanması gerekiğinde kümeye parantezlerinin ({ }) kullanılması zorunluluğu da aynı benzerlidir.

Javascript kodlarını var olan sözdizim kurallarına uygun biçimde yazma işlemi başta zor olabilir. Javascript'i ilk defa öğrenenler, HTML etiketlerini yazarken yaşadıkları esnekliğin Javascript'te olmadığını fark edeceklerdir. Örneğin HTML etiketlerinde kapatılması gereken bir etiket kapatılmadığında tarayıcılar bunu yorumlama aşamasında tolere ederek sanki kapanış etiketi varmış gibi davranışabilirler. Ayrıca HTML etiketleri büyük veya küçük harfe duyarlı değildir. Bu nedenle çoğu geliştirici, HTML dilini yazarken Javascript ile kodlamaya geçiklerinde sözdizim hataları ile çok fazla uğraşmak zorunda kalırlar.

Sözdizim hatalarını en aza indirmek için size tavsiyemiz, öncelikle çeşitli kaynaklardan gördüğünüz kodları bire bir olarak çalıştmayı denemenizdir. Ardından çalışan kodlarda değişiklik yaparak testler yapmanızdır. Bu sayede olası bir hatanın tespiti kolay olacaktır.

Yazım hatalarından kaçınmak için bir başka tavsiye ise Javascript diline destek veren editör kullanmaktır. Önceki üitede belirtilen editörlerin tümü bu konuda size yardımcı olacak özelliklere sahiptir. Bu özellikler;

- Komutları hatırlatma,
- Yanlış yazılan komutların altını çizme,
- Kapatılması unutulan parantezler olduğunda uyarı verme,
- İsimlendirme kurallarına uymayan durumlarda uyarı verme,
- Nesne tabanlı programlamada nesne fonksiyon ve üyelerini hatırlatma gibi sıralanabilir.

Bu üitede, Javascript ile kodlamaya başlayacağız. Bunun için dili oluşturan değişkenler, operatörler ve koşul ifadelerini ele alacağız. Her programlama dilinde ana yapı taşıları olarak kullanılan bu öğeler, hiç programlama bilmeyenler için belki zor gelebilir, ancak komutları olabildiğince HTML sayfalarına hitap edecek biçimde sunmaya çalışacağız. Bu noktada tavsiyemiz, gösterilen konuların mutlaka seçilecek bir editör ile bire bir test edilmesi ve ondan sonra geliştirilmeye çalışılmasıdır.

JAVASCRIPT VE HTML ETİKETLERİ İLİŞKİSİ

Üitede ele alınacak konuların HTLM sayfalarının içerisindeki diğer içerik veya etiketler ile iletişim hâlinde olması için, Javascript ile HTML etiketleri arasındaki bağlantıyı gösteren basit komutlar ile başlayacağız.



Sözdizim, yani yazım, hataları, çoğu editör tarafından kodu yazar yazmaz algılanabilir.

HTML Etiketlerine Javascript ile Erişmek

Javascript dili ile HTML sayfalarındaki tüm etiketlerin özelliğini veya içeriği değiştirmek mümkündür. Bunun için `document.getElementById()` komutu kullanılır.

Tablo 10.1. `document.getElementById()` fonksiyonu



document.getElementById() komutundaki harflerin büyük veya küçük olması önemlidir, kodlama yaparken buna dikkat etmelisiniz.

HTML ve Javascript Komutları
<pre><!DOCTYPE html> <html lang="tr"> <head> <meta charset="UTF-8"> <title>Ünite 8</title> </head> <body> <h1>Sayfa Başlığı</h1> <p id="yazi1">Bu bir paragraftır.</p> <script> document.getElementById("yazi1").innerText="Bu yazı javascript ile değiştirilmiştir."; </script> </body> </html></pre>
Web Sayfası Görüntüsü

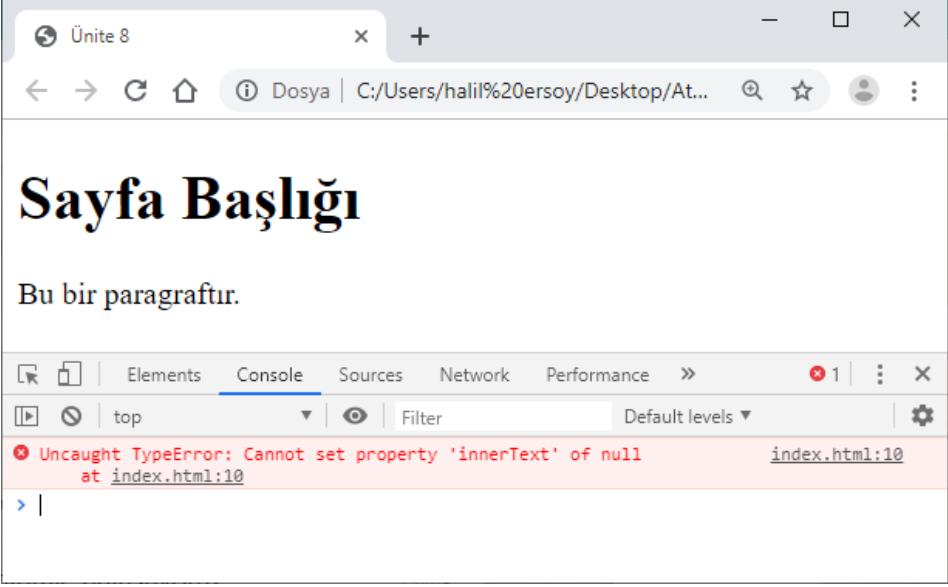
Yukarıdaki kod ve çıktısını incelediğimizde, Javascript komutları ile bir paragrafin *icerisinde* yazan yazının değiştirildiğini göreceksiniz. Bu komutun doğru şekilde çalışması için aşağıdaki koşulların sağlanmış olduğuna dikkat ediniz:

- HTML etiketlerine Javascript komutları ile erişmek için “*id*” özelliği ile *benzersiz* bir isim verilmelidir. Bu isim *Javascript isimlendirme kurallarına* uymalıdır. Aynı HTML sayfası içerisinde *aynı id’ye* sahip başka eleman *olmamalıdır*.
- `document.getElementById()` fonksiyonunun parantezleri içerisinde bu “*id*” bilgisi *çift tırnak/tek tırnak* içerisinde yazılmalıdır.
- `document.getElementById()` komutu, *etiket yüklenikten sonra çalışabilir*, öncesinde çalışmaz.

Özellikle yukarıdaki son koşulu iyi anlamak gereklidir. Etiketlere ulaşmak için kullanılacak Javascript komutları, *etiket yüklenikten sonra* bu erişimi

sağlayabilir. Aksi durumda etiket bulunamaz ve **çalışma hatası** verilir. Bu olası hatalı durumu göstermek için aşağıdaki kodları ve çıktıyı inceleyelim.

Tablo 10.2. Mantık Hatası

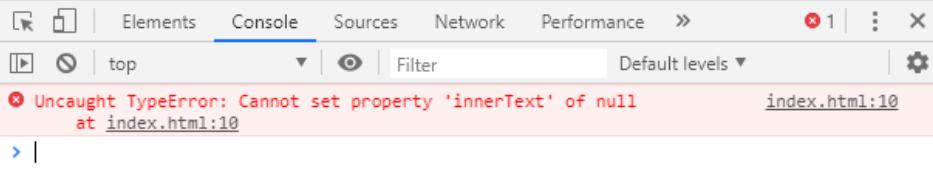
HTML ve Javascript Komutları	
<pre><!DOCTYPE html> <html lang="tr"> <head> <meta charset="UTF-8"> <title>Ünite 8</title> </head> <body> <h1>Sayfa Başlığı</h1> <script> document.getElementById("yazi1").innerText="Bu yazı javascript ile değiştirilmiştir."; </script> <p id="yazi1">Bu bir paragraftır.</p> </body> </html></pre>	
Web Sayfası Görüntüsü	
	



Çoğu hata mesajı size anlamsız gelebilir, ancak hatanın olduğu satırın numarasını görmeniz, hatayı anlamanızı kolaylaştırır.

Sayfa Başlığı

Bu bir paragraftır.



```
Elements Console Sources Network Performance
✖ 1 | ⋮ ×
top | Filter Default levels ▾
✖ Uncaught TypeError: Cannot set property 'innerText' of null
at index.html:10
> |
```

Yukarıdaki örnekte Javascript kodu, paragraf etiketinin içeriğini değiştirememiştir. Bunun nedeni, Javascript kodunun **çalıştığı sırada** henüz ilgili paragraf etiketi **yüklenmemiştir**. Bu nedenle tarayıcıda görülen hata mesajı ortaya çıkmıştır (Tarayıcının tespit ettiği hataları göstermesi için **F12** tuşuna basarak **Geliştirici seçeneklerini** ve **Console** sekmesini açmalısınız).

Örnekteki “**innerText**” özelliği, **p**, **h1** (tüm başlık etiketleri), **div**, **span**, **td** gibi **icerisinde metin barındıran tüm etiketler** için kullanılabilir.

Basit Mesajlar Yazdırma

Ünitenin devamındaki konularda birçok işlem sonucunda ortaya çıkacak çıktıların görülebilmesi için aşağıdaki 3 basit komut ile **yazdırma** işlemi yapılacaktır. Bunlar;

- alert();
- document.write();
- console.log();

komutlarıdır.

Yukarıdaki 3 komutun çalışmasını gösteren örnek uygulamayı aşağıdaki kodlar ve çıktıda görebilirsiniz.

Tablo 10.3. Mesaj Veren 3 Basit Komut Ve Çıktıları

HTML ve Javascript Komutları
<pre><body> <h1>Sayfa Başlığı</h1> <p id="yazi1">Bu bir paragraftır.</p> <script> document.getElementById("yazi1").innerText="Bu yazı javascript ile değiştirilmiştir"; console.log("bu console.log() mesajıdır"); document.write("Bu document.write() mesajıdır"); alert("bu alert() mesajıdır"); </script> </body></pre>
Web Sayfası Görüntüsü
Console Output
<pre>Ünite 8 Dosya C:/Users/halil%20ersoy/Desktop/Atatürk%20Üniversi... Sayfa Başlığı Bu yazı javascript ile değiştirilmiştir bu alert() mesajıdır Bu document.write() bu console.log() mesajıdır index.html:12</pre>



Alert() komutunun sonucunda açılacak küçük pencere, farklı tarayıcılarda farklı görünecektir, bu normaldir.

Kodlardaki **alert()** fonksiyonu, parantezi içine yazdığınız metin ifadelerini veya değişkenlerin değerlerini tarayıcı ekranına açılacak yeni küçük bir pencere içerisine yazar. Bu pencere, üzerindeki “Tamam” butonuna basılıncaya kadar bekler ve kapanmadığı sürece sayfada başka bir işlem yapılmasına *izin vermez*.

Komutlardan **document.write()**, parantezleri içerisinde yazacağınız açık metinleri veya değişkenlerin değerlerini *çalıştığı yere* HTML içeriği olarak yazar. Benzer bir komut da **document.writeln()** komutudur. Önceki komuttan tek farkı HTML sayfasının içerisinde yazının sonuna *satır sonu karakteri* eklemek ve kendisinden sonra gelen yazıların *kodda* alt satırдан başlamasını sağlamaktır. Ancak unutulmamalıdır ki HTML *kodlarında* alt alta yazılmış ifadeler, tarayıcı tarafından *ekrana* alt alta yazılmaz! Bunun için HTML etiketlerinden **
** ifadesinin eklenmesi gereklidir.

Son olarak ***console.log()*** komutu, ekrana değil, tarayıcının (eğer açıksa) ***geliştirici seçeneklerindeki konsol ekranına*** çıktı yazmak üzere kullanılır.

Parantezleri içersine metin ya da değişken yazılarak konsol ekranına çıktı gönderilebilir. Bu komut daha çok geliştiricilerin birtakım değerleri veya işlemleri ***takip etmek*** için tercih ettikleri yoldur. Bu sayede sonradan silinecek birtakım çıktıların sayfanın içerisinde yer alarak unutulması riskini yok etmiş olurlar.



Örnek

- Alert, document.write ve console.log komutları içerisinde birden fazla değer ve değişkeni birlikte yazabilirsiniz. Bunun için yazıları tırnak içerisinde, sayısal değerleri olduğu gibi ve değişkenleri de olduğu gibi aralarına + işaretini koyarak birleşik şekilde kullanabilirsiniz.
- `•alert("Sonuç = " + 2 + "olur.");`
- **ÇIKTI:** Sonuç 2 olur.

Açıklama Satırları

Geliştiriciler, kodlama işlerini yaparken kendileri veya başka geliştiricilere hatırlatma yapmak, bilgi vermek ya da geçici olarak bazı komutları devre dışına almak için kodlarına ***açıklama*** satırları eklerler. Bunu;

- Eğer *tek bir satır* açıklama satırı eklenecek ise, başına // sembollerini yazarak;
- Eğer *birden fazla satır* toplu hâlde açıklama olarak eklenecek ise başına /* ve sonuna */ sembollerini koyarak yapabilirler.

Açıklama satırları ***çalıştırılmaz*** ancak kodlarla birlikte kaydedilebilir.

Aşağıdaki örnekteki açıklama satırlarını ve programın çıktısını inceleyiniz.



Geçici olarak çalışmasını aşağıya almak istediğiniz komutları, satır başına // işaretini koyarak açıklama satırı hâline getirebilirisiniz.

Tablo 10.4. Açıklama Satırları Ve Çıktısı

Javascript Komutları

```
document.write("<p>satır 0</p>");
// aşağıdaki bazı satırlar açıklama olarak işaretlenmiştir
// ve çalışmazlar
// document.write("<p>satır 1</p>");
document.write("<p>satır 2</p>");
/*
document.write("<p>satır 3</p>");
document.write("<p>satır 4</p>");
*/
document.write("<p>satır 5</p>");
```

satır 0

satır 2

satır 5

DEĞİŞKENLER

Her programlama dilinde olduğu gibi, Javascript'te de değişken tanımlama ve kullanma önemli bir gerekliliktir. **Değişken**, programın çalışması sırasında birtakım değerleri (sayı, yazı, tarih, vb.) bilgisayarın hafızasında (RAM) saklamaya yarayan hafıza alanlarına verilen isimlerdir. **Değişken adı** kullanıcı tarafından belirlenir. Değişken tanımlamak için **var** komutu kullanılır.



Değişken isimlendirme kuralları bir önceki üniteye anlatılmıştır.

Tablo 10.5. Değişken Tanımlama

Javascript Komutları	
<script>	<pre> var a; var b=50; var c=3.14; var d="merhaba"; var e="dünya"; f=80; document.write("a=" + a + "
"); document.write("b=" + b + "
"); document.write("c=" + c + "
"); document.write("d=" + d + "
"); document.write("e=" + e + "
"); document.write("f=" + f + "
"); </script> </pre>
Web Sayfası Görüntüsü	
	<pre> a=undefined b=50 c=3.14 d=merhaba e=dünya f=80 </pre>

Yukarıdaki örnekte toplam 6 adet değişken tanımlanmıştır. Ardından değişkenlerin değerleri ekrana yazılmıştır. Yukarıdaki tanımlamalar ve çıktılar incelendiğinde aşağıdaki değişken tanımlama kuralları anlaşılabilir:

Tablo 10.6. Değişken Tanımlama Kuralları

Kural	Örnek ve Çıktı
1. Değişkenler var komutu ile tanımlanır, istenirse tanımlama sırasında ilk değer verilebilir.	var b; b=50; YA DA var b=50;
2. İlk değeri verilmeyen değişkenlerin değeri "undefined" olur.	var a; document.write(a); → undefined
3. Aynı var ifadesi ile birden çok	Var a,b,c=30; document.write(a); → undefin .

değişken tanımlanabilir. Yandaki örnekte sadece c değişkeninin değeri 50 olur, diğerlerinin "undefined" olur.	document.write(b); → undefined document.write(c); → 50
4. Var komutu ile aynı isimde bir başka değişken tekrar tanımlanabilir, bu durumda önceki değişken yok edilmiş olur.	var a=20; document.write(a); → 20 var a = 75; document.write(a); → 75
5. Aynı satırda birden fazla değişkene aynı ilk değer verilebilir.	var a=b=c=40; document.write(a); → 40 document.write(b); → 40 document.write(c); → 40
6. Bir <script> blokunda tanımlanan değişken, bu blok kapandıktan sonra açılacak bir sonraki script blokunda da kullanılabilir.	<script> var a=30; </script> <p>yazı</p> <script> document.write(a); → 30 </script>



Değişkenleri kullanılabacakları fonksiyon ya da blok içerisinde ilk sırada tanımlamak iyi bir alışkanlıktır.

Göründüğü üzere, değişkenler konusunda Javascript oldukça esnektir. Hatta bir değişken **var** kelimesi kullanılmadan da ilk değer ataması ile tanımlanmış olur. Yukarıda Tablo 10.5'teki "f" değişkeni buna bir örnektir. Ancak bu durumda "f" değişkeni **global** bir değişken olacaktır, bu nedenle bu şekilde kullanım önerilmez. Değişkenlerin **global** veya **lokal** olması, ileriki bölümlerde **Fonksiyonlar** başlığında açıklanacaktır.

Bu esnekliğe rağmen, geliştiricilerin sayfada kullanacakları değişkenleri diğer Javascript komutlarından önce **var** komutu ile tanımlamaları önerilir.

Değişkenler tanımlanmadan, herhangi bir işlemde kullanılamazlar. Aşağıdaki örnekte **a** ve **b** değişkenleri tanımlanmış ancak **c** değişkeni tanımlanmadığı için **yazım hatası** oluşacaktır.

Tablo 10.7. Tanımlanmamış Değişken Kullanılamaz.

Javascript Komutları
<pre><script> var a=50; document.write(a); → 50 b=40; document.write(b); → 40 document.write(c); → HATA !! </script></pre>

Değişken ve Veri Türleri

Değişkenlerin içerisinde saklanacak veri çeşidine göre **türü** vardır. Farklı turdeki değişkenleri tanımlamak için yine **var** ifadesi kullanılır, ancak bundan sonra değişkene atanacak **ilk değerin türüne göre**, değişken türü belirlenmiş olur.

Tablo 10.8. Farklı Veri Türleri

Javascript Komutları
<pre><script> var a; // tür : belirsiz (undefined) a=50; // tür: sayı (number) var b=3.14; // tür: sayı (number) var c="merhaba"; // tür: metin (string) var d= 123e5; // tür: bilimsel sayı (123 x 10⁵) (number) var e='t'; // tür: metin (string) </script></pre>

Yukarıdaki Tablo 10.8.'de görüleceği üzere farklı biçimlerde değerleri tutan değişkenlerin **number** veya **string** türünde olduklarını gördük. Örnekteki "a" değişkeni, ilk değer olarak 50 değerini alana kadar türü **belirsiz (undefined)** olur.

Tür Dönüşümü

Değişkenlere farklı türde değer vererek o değişkenin **türünü değiştirmiş** oluruz. Aşağıdaki örnekte aynı değişkene atanın farklı değerlerin tür değişikliği etkisini görebilmek için, **typeof()** komutu ile değişkenin türünü ekrana yazdırıyoruz.

Tablo 10.9. Farklı değerler ile tür dönüşümleri ve **typeof** komutu

Javascript Komutları
<pre>var a=3.14; document.write(typeof(a)); → number a = "merhaba"; document.write(typeof(a)); → string a=123e5; document.write(typeof(a)); → number a='r'; document.write(typeof(a)); → string a=30; document.write(typeof(a)); → number</pre>

True ve false kelimeleri Javascript'in anahtar kelimelerindendir, metin türü ile karıştırılmamalıdır.



Düzenleme

Javascript'te **number** (sayı) ve **string** (metin) türleri temel veri türleridir. Sayı ve metin türündeki değerlerin grup hâlinde kullanılmasıyla farklı veri türleri ortaya çıkmıştır. Bunlar **boolean** (ikilik) ve **object** (nesne) türleridir.

Boolean veri türü, değer olarak **true** ve **false** ifadeleri tutan değişkenler için kullanılır. Bu tür değişkenler, herhangi bir kıyaslama işleminin sonucunu **doğru (true)** ya da **yanlış (false)** olarak tutabilirler.

Tablo 10.10. Boolean Veri Türü

Javascript Komutları
<pre>var a= 4; var b= 8; var c= b > a; document.write("c = " + c + " tür: "+ typeof(c)); Çıktı c = true tür: boolean</pre>

Object veri türü, aynı veya farklı türdeki birden çok değişkenin gruplanması gerekiğinde başvurulan bir türdür. Bu türe yönelik ilk örnek **dizilerdir** (**array**). Aşağıdaki örnekte dizi tanımlaması ve kullanımı gösterilmiştir. Ancak daha detaylı dizi kullanımı bir sonraki ünite de **Diziler** başlığı altında anlatılmıştır.

Tablo 10.11. Dizilerin Türü Object'tır.

Diziler Javascript'te çok kullanılmasına rağmen tür olarak nesne (object) sınıfındadır.

Javascript Komutları	
var a=[10,20,30,40,50]; var c= a[0] + a[1]; // 20 ve 30 toplanıyor document.write("c = " + c + " tür: " + typeof(c)); document.write("a[0] = " + a[0] + " tür: " + typeof(a[0])); document.write("a dizisi tür: " + typeof(a));	Çıktı
c = 50 tür: number a[0] = 10 tür: number a dizisi tür: object	

Farklı türleri gruplayarak da **object** türünde değişkenler (**nesneler**) yaratılabilir.

Tablo 10.12. Farklı Türlerden Oluşan Nesne Türü

Javascript Komutları	
var insan = {ad: "ali", soyad:"tok",yas:50, gozRengi:"mavi"}; document.write("insan.ad=" + insan.ad + " tür: " + typeof(insan.ad)); document.write("insan tür: " + typeof(insan));	Çıktı
insan.isim=ali tür: string insan tür: object	

Yukarıdaki son iki örnekte görüleceği üzere, **nesne** türündeki değişkenlerin (a ve insan) kendi türleri **object** iken, nesnenin parçalarının her biri kendi temel türündedir (a[0] elemanı **number**, insan.isim elemanı **string** türündedir).

Nesne (object) veri türü detayları bu kitabın kapsamına alınmamıştır.

Değişken Türlerinin Önemi

Yukarıda tanıtılan değişken türlerinden uygun olanı seçmemiz gereklidir. Bunun iki sebebi vardır:

- Bazı veri türleri hafızada (kullanıcının bilgisayarının RAM belleğinde) diğerlerine göre daha fazla yer işgal eder. Gereksiz yere büyük veri türleri seçmek, kullanıcının bilgisayarını gereksiz yere yoracaktır.
- Verinin türüne göre, veri üzerinde yapılacak işlemler farklılık gösterir, uygun olmayan türlerde yapılan işlemler yanlış sonuçlara neden olur.

Özellikle ikinci sebebi aşağıdaki örnek ile gösterelim.

Tablo 10.13 Toplama İşleminin Farklı Türlerdeki Etkisi

Javascript Komutları	
var a=10, b=20, c;	
var x="10", y="20", z;	
c= a + b;	
z = x + y;	
document.write("c=" + c + "</br>");	
document.write("z=" + z + "</br>");	
	ÇIKTI
c=30	
c=1020	

Toplama işlemi için kullanılan artı operatörü (+), işlem yapmadan önce kendisinin sağındaki ve solundaki değer ya da değişkenlerin türlerine bakar ve türe göre işlem yapar. Eğer her iki taraf da *sayı* ise *matematiksel toplama* yapar ($10 + 20 = 30$). Eğer taraflardan en az bir tanesi *metin* (string) ise toplama yerine *bitiştirme* yapar (“10” + “20” = “1020”). Bu tür işlemlerin beklentiği gibi çalışması için işleme giren değişken ya da değerlerin türleri doğru oluşturduğumuzda emin olmalıyız.



Toplama ve bitiştirme operatörü olan + simbolü, sonucu doğru vermesi için doğru türdeki değişken ya da değerler ile çalıştırılmalıdır.

Undefined ve Null Veri Değerleri

Değişkenler henüz ilk değerini almadıklarında *tür* ve *değer* olarak *undefined* şeklinde tanımlanır. Herhangi bir değişken açık biçimde *undefined* değerine atanarak *hem içerisindeki değer* silinmiş olur, *hem de türü undefined olarak sıfırlamış* olur.

Tablo 10.14. Undefined Değeri

Javascript Komutları	
var m;	
document.write("m=" + m + " tür:" + typeof(m) + "</br>");	
m=100;	
document.write("m=" + m + " tür:" + typeof(m) + "</br>");	
m=undefined;	
document.write("m=" + m + " tür:" + typeof(m) + "</br>");	
m="merhaba";	
document.write("m=" + m + " tür:" + typeof(m) + "</br>");	
	ÇIKTI
m=undefined tür: undefined	
m=100 tür: number	
m=undefined tür: undefined	
m=merhaba tür: string	

Javascript'te *null* değeri ise nesnelerin değerini yok etmek için kullanılır. *Null* değerini alan nesnenin içindeki *değer silinirken*, türü yine *object* olarak kalır. Eğer nesnenin hem değerini hem de türünü sıfırlamak istenirse *undefined* değerine atanmalıdır.

Tablo 10.15. Undefined Ve Null Değerleri

Javascript Komutları	
var m=[10,20,30,40];	
document.write("m=" + m + " tür:" + typeof(m) + "</br>");	
m=null;	

```

document.write("m=" + m + " tür:" + typeof(m) + "</br></br>");

var m=[10,20,30,40];
document.write("m=" + m + " tür:" + typeof(m) + "</br>");
m=undefined;
document.write("m=" + m + " tür:" + typeof(m) + "</br>");

ÇIKTI

m=10,20,30,40   tür: object
m=null          tür: object

m=10,20,30,40   tür: object
m=undefined     tür: undefined

```

OPERATÖRLER

Değişkenlerin değerlerini değiştirmeye veya değerlerini kontrol etmeye yarayan komutlara operatör denir. Operatörler genellikler sembollerden oluşur. Yaptıkları işler bakımından operatörleri 3 sınıfta görelim:

- Aritmetik operatörleri,
- Karşılaştırma operatörleri,
- Mantık operatörleri.



Aşağıdaki 3 komut eşittir:
 $a=a+1;$
 $a+=1;$
 $a++;$

Aritmetik Operatörleri

Sayısal değişkenler üzerinde yapılabilecek aritmetik işlemlerini gerçekleştiren aritmetik operatörleri aşağıdaki tabloda verilmiştir. Tablodaki işlemler **a**, **b** ve **c** değişkenlerinin ilk satırdaki tanımlamalarına göre yapılmıştır.

Tablo 10.16. Aritmetik Operatörleri

Operatör	Anlamı	Örnek <code>var a=10, b=20, c;</code>	Sonuç c
+	Toplama	<code>c = a + b;</code>	30
-	Çıkarma	<code>c = a - b;</code>	-10
*	Çarpma	<code>c = a * b;</code>	200
/	Bölme	<code>c = a / b;</code>	0.5
%	Bölmünden kalan (mod)	<code>c = a % b;</code>	10
++	Sayıyı bir artırmak	<code>c = 5; c++; veya ++c;</code>	6
--	Sayıyı bir azaltmak	<code>c = 5; c--; veya --c;</code>	4

Yukarıdaki operatörlerden **artırma (++)** ve **azaltma (--)** operatörleri, kullanım şekline göre artırma ve ya eksiltme işlemini, atamadan **önce** ya da **sonra** yapar. Aşağıdaki şekli inceleyelim.

Tablo 10.17. Artırma Ve Azaltma Operatörünün Farklı Kullanımı

İşlemler	Çıktı
<code>var a=5, b=1; var sonuc; sonuc = a + b++;</code>	

<code>document.write("sonuc=" + sonuc + "</br>"); document.write("a=" + a + "</br>"); document.write("b=" + b + "</br>");</code>	sonuc=6 a=5 b=2
<code>var a=5, b=1; var sonuc; sonuc = a + ++b; document.write("sonuc=" + sonuc + "</br>"); document.write("a=" + a + "</br>"); document.write("b=" + b + "</br>");</code>	sonuc=7 a=5 b=2

Yukarıdaki örnekte **sonuç** değişkeninin değerinin farklı olmasının nedeni **artırma** (++) operatörünün değişkenin **solunda** ya da **sağında** yazılmasından kaynaklanmaktadır. Eğer ilk örnekteki gibi değişkenin **sağına yazılırsa** (b++), toplama işlemine değişkenin artmamış değeri (1) girer ve sonuç 6 değerine ulaşır. Ardından **b** değişkeni 1 artar ve 2 olur. İkinci örnekte ise artırma operatörü değişkenin soluna (++b) yazılmıştır. Bu durumda toplama işleminden önce artırma yapılır (**b** değişkeni 2 olur) ve sonuç 7 olarak hesaplanır.

Bazı aritmetik işlemler, **değişkenin kendisini kullanarak** işlem yapıyorsa, bu işlemleri aşağıdaki tablodaki gibi **kısaltarak** kullanmak mümkündür.

Tablo 10.18. Aritmetik İşlemlerin Kısaltılmış Hali

Operatör	Anlamı	Örnek ve Alternatif	Sonuç
<code>+=</code>	Kendisiyle bir değeri topla	<code>a += b; // a=a+b;</code>	<code>a = 30</code>
<code>-=</code>	Kendisinden bir değeri çıkar	<code>a -= b; // a=a-b;</code>	<code>a = -10</code>
<code>*=</code>	Kendisiyle bir değeri çarp	<code>a *= b; // a=a*b;</code>	<code>a = 200</code>
<code>/=</code>	Kendisini bir değere böl	<code>a /= b; // a=a/b;</code>	<code>a = 0,5</code>
<code>%=</code>	Kendisinin bir değere bölümünden kalanı (mod) bul	<code>a %= b; // a=a%b;</code>	<code>a = 10</code>

Karşılaştırma Operatörleri

Değişken ya da değerlerin birbiri ile kıyaslanması için kullanılan operatörlere **karşılaştırma operatörleri** denir. Bu operatörler belirli noktalarda karar vermeye ve gerekirse programın akışını değiştirmek için, az sonra ayrı bir başlık ile anlatılacak olan, koşul yapıları için gereklidir. Karşılaştırma operatörlerinin sonucunda **true (doğru)** ya da **false (yanlış)** değerleri üretilir. Bu değerler metin ya da sayı değildir, türü **boolean**'dır. Javascript'teki karşılaştırma operatörlerini aşağıdaki tabloda görebilirsiniz.

Tablo 10.19. Karşılaştırma Operatörleri

Operatör	Anlamı	Örnek	Sonuç
<code>==</code>	Değerleri birbirine eşitse	<code>a == 2</code>	<code>false</code>
		<code>a == 10</code>	<code>true</code>
<code>!=</code>	Değerleri eşit değilse	<code>a != 9</code>	<code>true</code>
<code>==</code>	Değerleri ve türleri eşitse	<code>a === "10"</code>	<code>false</code>
		<code>a === 10</code>	<code>true</code>



Kısaltma operatörlerini yazarken, iki symbol arasında boşluk bırakmak hata olacaktır.
`a += 3; // doğru`
`a + = 3; // hatalı`



En çok yapılan hata, atama operatörü (=) ile eşitlik kontrolü operatörünün (==) karıştırılmasıdır.

!==	Değerleri ve türleri eşit değilse	a != "10"	true
>	Değeri büyükse	a > 8	true
<	Değeri küçükse	a < 8	false
>=	Değeri büyük veya eşitse	a >= 10	true
<=	Değeri küçük veya eşitse	a <= 9	false

Yukarıdaki operatörlerden **==** ve **!=** operatörleri, sağındaki ve solundaki değişken veya değerlerin *hem değerine hem de türüne göre* karar verir. Buna göre değerleri aynı, ancak türleri farklı değişkenlerin karşılaştırılmasındaki sonuçları aşağıda görebilirsiniz.

Tablo 10.20. Değer Karşılaştırma Ve Tür Karşılaştırma Operatörleri

Javascript Komutları			
Örnek ve Çıktılar			
Karşılaştırma	Sonuç	Türler	Karşılaştırma Ölçütü
a == b	true	number ve number	Sadece değer eşit mi?
a == c	true	number ve string	
b == c	true	number ve string	
a === b	true	number ve number	Hem değer hem tür eşit mi?
a === c	false	number ve string	
b === c	false	number ve string	
a != b	false	number ve number	Sadece değer farklı mı?
a != c	false	number ve string	
b != c	false	number ve string	
a !== b	false	number ve number	Hem değer hem tür farklı mı?
a !== c	true	number ve string	
b !== c	true	number ve string	

Mantık Operatörleri

Birden fazla karşılaştırma işleminin birlikte değerlendirilmesi ve tek bir doğru (**true**) ya da yanlış (**false**) üretilebilmesi için mantık operatörleri kullanılır. Bu operatörler típkı karşılaştırma operatörleri gibi, koşul ve döngü yapılarında sıkılıkla kullanılırlar. Mantık operatörleri aşağıdaki tabloda verilmiştir.

Tablo 10.21. Mantık Operatörleri

Operatör	Anlamı	Örnek var a=10, b=20;	Sonuç
&&	ve	a <= 10 && b < 100	true
 	veya	a == 5 b == 5	false
!	değil	!(a > b)	false


&& ve **||** operatörlerinin tek sembollük hâlleri de vardır (**&** ve **||**). Ancak onlar farklı işlevleri yerine getirmektedirler ve bu kitabın kapsamında değildir.

Mantık operatörleri iki ya da daha fazla karşılaştırma ifadesini karşılaştırabilir. Buna göre;

- *Ve (&&)* operatörünün *her iki yanında true* değeri varsa sonuç *true*, bunun dışındaki tüm durumlarda sonuç *false* olur.
- *Veya (||)* operatörünün yanındaki ifadelerden *en az bir tanesi true* ise sonuç *true*, tüm ifadeler *false* ise sonuç *false* olur.
- *Değil (!)* operatörü sağ tarafına *tek ifade* alır ve bu ifadenin *boolean* cinsinden *tersini* (*true* ise *false*, *false* ise *true*) üretir.

Operatörlerde İşlem Önceliği

Yukarıdaki operatörler kendi başlarına iki ya da tek değişken ile işlem yaparlar. Buna karşın bazı işlemler *birden fazla operatörün* sırayla kullanılmasını gerektirebilir. Bu durumda farklı operatörlerin *yan yana* yazılarak sonucun üretilmesi gerekebilir. Böyle durumlarda yan yana yazılmış operatörlerin hangisinin *öncelikle* çalışacağı bilinmelidir. Aşağıdaki tabloda *operatörlerin işlem önceliği* belirtilmiştir.



İşlem önceliği aynı düzeyde olan operatörlerin işlem yönüne bakılmalıdır.

Tablo 10.22 Operatör İşlem Önceliği Ve Yönü

Operatör	İşlevi	İşlem Önceliği	İşlem Yönü
<code>++</code>	Değeri 1 artırma	Birinci	Sağdan sola (\leftarrow)
<code>--</code>	Değeri 1 azaltma	Birinci	Sağdan sola (\leftarrow)
<code>-</code>	Eksi değer	Birinci	Sağdan sola (\leftarrow)
<code>!</code>	Değil	Birinci	Sağdan sola (\leftarrow)
<code>* , / , %</code>	Çarpma, bölme, mod alma	İkinci	Soldan Sağa (\rightarrow)
<code>+ , -</code>	Toplama, çıkarma	Üçüncü	Soldan Sağa (\rightarrow)
<code>+</code>	Bitiştirme	Üçüncü	Soldan Sağa (\rightarrow)
<code>< , <=</code>	Küçük, küçük veya eşit	Dördüncü	Soldan Sağa (\rightarrow)
<code>> , >=</code>	Büyük, büyük veya eşit	Dördüncü	Soldan Sağa (\rightarrow)
<code>==</code>	Eşit	Beşinci	Soldan Sağa (\rightarrow)
<code>!=</code>	Farklı	Beşinci	Soldan Saşa (\rightarrow)
<code>====</code>	Tür ve değer aynı	Beşinci	Soldan Saşa (\rightarrow)
<code>!==</code>	Tür ve değer farklı	Beşinci	Soldan Saşa (\rightarrow)
<code>&&</code>	Ve	Altıncı	Soldan Saşa (\rightarrow)
<code> </code>	Veya	Yedinci	Soldan Saşa (\rightarrow)
<code>=</code>	Atama	Sekizinci	Sağdan sola (\leftarrow)
<code>+= , -= , *= , /= , %=</code>	Aritmetik işlem ve atama	Sekizinci	Sağdan sola (\leftarrow)

Yukarıdaki tabloda operatörler işlem önceliğine göre sıralanmıştır. Farklı operatörler yan yana yazıldığından yukarıdaki *önceliğe* göre işlem yapmaya başlanır. Eğer önceliği eşit olan operatörler yan yana yazılırsa, o zaman *işlem yönüne göre* öncelik verilir.

İşlem önceliği değiştirmek istenirse, öncelik verilmek istenen operatör ve o operatör için gerekli değişken ve değerler **parantez ()** içeresine alınır.

Tablo 10.23. İşlem Önceliği Örneği

Javascript Komutları ve Örnek Çıktılar
<pre>var a=5, b=1, c=4; var sonuc; sonuc = a + b / c; → 5.25 // öncelik bölmede sonuc = (a + b) / c; → 1.5 // öncelik toplamada sonuc = a / c / 2; → 0.625 // öncelik soldan sağa</pre>

Mantık ve karşılaştırma operatörleri, eğer aritmetik işlemler ile birlikte kullanılacak ise, **öncelik** aritmetik işlemlerdedir.

Tablo 10.24 İşlem Önceliği Ve Yönü Örneği

Javascript Komutları ve Örnek Çıktılar
<pre>var a=5, b=1, c=4; var sonuc; sonuc = a > b && a > c; → true //öncelik karşılaştırmada (>) sonuc = a % 2 == 0; → false //öncelik mod almada (%) sonuc = a >= 2 - 1 && b >= 0 && c >= 0; → true // öncelik çıkarmada (2 - 1), sonra karşılaştırmalarda (>=), // ardından soldan sağa mantık operatörleri (&&)</pre>



İşlem önceliğini kontrol etmek için istenirse birden fazla parantez iç içe yazılabilir.
 $x=((a+1)/b)*c;$



İşlem önceliğini kontrol etmek için sadece düz parantezler kullanılır. Farklı parantez türleri hataya neden olur.
 $x=[(a+1)/b];$ // HATA
 $x=((a+1)/b);$ // Doğru



Bireysel Etkinlik

- Aşağıdaki komutların sonucunu önce kağıt üzerinde hesaplayınız. Ardından bilgisayarda kodlayıp sonuçları karşılaştırınız.
- $sonuc = a * a + b * b;$
- $sonuc = a * (a + b) * b;$
- $sonuc = a++ * b;$
- $sonuc = a + 1 \% 10;$

KOŞUL KOMUTLARI VE KOŞUL OPERATÖRÜ

Tüm programlama dillerinde komutların çalışma sırası yukarıdan aşağıyadır. Buna programın akışı denilebilir. Bu akış bazı durumlarda değişim zorundadır. Akışı belirli koşullara göre değiştiren komutlara **koşul komutları** denir. Javascript'te iki adet koşul komutu vardır:

- if
- switch

Bu iki komuta ek olarak, koşula göre değer aktarma işlemini değiştiren bir koşul operatörü vardır:

- ?:

If Komutu

Bu komut, kendisi içerisinde yazılan kıyaslama işleminin sonucunda **true** değeri üretilirse devamındaki komutları çalıştırır. Sonuç **false** ise devamındaki komutları çalışmaz.

Tablo 10.25. If Komutu Ve Çıktısı

Javascript Komutları
<pre>var a=5, b=6; if(a%2==0) // koşul false'dur document.write("a değişkeni çift sayıdır"); // çalışmaz if(b%2==0) // koşul true'dur document.write("b değişkeni çift sayıdır"); // çalışır ÇIKTI b değişkeni çift sayıdır</pre>

Yukarıdaki örnekte, **if** komutundan sonraki parantez içerisinde bir **koşul yazmak** mecburidir. Bu koşul ifadesi birden çok karşılaştırma ve mantık operatörünü içerebilir. Parantez içerisindeki tüm kıyaslamalar, varsa aritmetik işlemler ve mantık karşılaşmaları tamamlandıktan sonra ortaya çıkacak olan **true/false** değerine göre işlem yapılır.

If komutunun koşulu **doğru (true)** ise kendisinden sonraki bir komut çalıştırılır, **yanlış (false)** ise yine kendisinden sonraki bir komut çalıştırılmaz. Ancak bazı durumlarda bu koşula bağlı olarak birden çok komut veya komut satırı benzer şekilde çalışmalı ya da çalışmamalıdır. Bu durumda **if** komutunun etkisini birden çok komuta taşımak için, ilgili komutlar **küme parantezleri { }** içerisinde yazılmalıdır.

Tablo 10.26. Küme Parantezleri İle Blok Yaratmak

Javascript Komutları
<pre>var pi=3.1417, a=20; if(pi > 3.0 && a%2==0){ // koşul true'dur document.write("pi değeri 3ten büyükür"); // çalışır document.write("a değişkeni de çift sayıdır"); // çalışır } ÇIKTI Pi değeri 3ten büyükür a değişkeni çift sayıdır</pre>

If komutunun 3 farklı farklı biçimini vardır. Bunlar aşağıdaki gibidir.

Tablo 10.27. If Komutunu 3 Farklı Biçimi Ve Açıklaması

If Komutu Çeşitleri		
if(koşul) komutlar;	if(koşul) komutlar1; else komutlar2;	if(koşul1) komutlar2; else if(koşul2) komutlar2; else if(koşul3) komutlar3; else komutlar4;



if...else if komutunda, "else" ile if'i bitişik yazmak yanlıştır.
else if(koşul)//doğru
elseif(koşul) //hata

Koşul true ise, komutlar çalıştırılır	Koşul true ise sadece komutlar1 çalıştırılır, false ise sadece komutlar2 çalıştırılır.	Sırası ile ilk koşuldan itibaren kontrol edilir, sonucu true olan koşul bulunursa, sadece onun altındaki komutlar çalıştırılır ve devamındaki koşullara bakılmadan if komutu sonlandırılır. Eğer hiçbir koşul true değil ise, o zaman else komutunun altındaki komutlar çalıştırılır.
---	---	--

Yukarıdaki tabloda belirtilen “**komutlar**” ifadesi, eğer birden fazla komut çalıştırılmak isteniyorsa, **küme parantezi** içerisine alınmış komut blokları ile değiştirilmelidir. Eğer tek komut çalıştırılacak ise küme parantezine almayı gerek yoktur, ancak alınırsa da hata olmaz.

Tablo 10.28. İf.. Else İf...Else Örneği

Javascript Komutları	
var a=5, b=10, c=4;	
<pre>if(a%2==0) // false document.write("a değişkeni çift sayıdır");// çalışmaz else if(b%2==0) // true document.write("b değişkeni çift sayıdır");// çalışır else if(c%2==0) // kontrol edilmez document.write("c değişkeni çift sayıdır");// çalışmaz else document.write("hiç biri çift değildir"); // çalışmaz</pre>	
ÇIKTI	
b değişkeni çift sayıdır	

Yukarıdaki örnekte, ikinci koşul ($b \% 2 == 0$) **true** olduğu için, ikinci komut çalışır. Bu noktadan sonra artık **kontrol yapılmaz** ve **if** komutu sonlanır. Dikkatlice incelendiğinde, aslında üçüncü komutun da **true** olduğu görülebilir, ancak **if...else if...else** yapısında, herhangi bir koşul doğru ise sadece ona bağlı komutlar çalıştırılır ve diğer koşullar kontrol edilmez.

If komutu, içerisinde
elseif ve else içermek
zorunda değildir.



Bu biçimde kullanılan **if** komutunda istenilen sayıda **else if() koşulu** ve komutları eklenebilir. İstenirse **else** ifadesiyle, önceki tüm koşulların doğru olmadığı durumlarda çalışması için komut veya komutlar eklenebilir, ancak **else** ifadesi mutlaka **en sonda** ve **sadece bir defa** yazılmalıdır. Eğer istenirse, **else** ifadesi hiç yazılmayabilir.

Aşağıdaki örnekte **birbirinden bağımsız üç if komutu** görülmektedir. Bu şekilde bağımsız olan **if** komutlarında, tüm koşullar kontrol edilir ve doğru olan tüm koşulları için ilgili komutlar çalıştırılır.

Tablo 10.29. Bağımsız Üç If Komutu

Javascript Komutları	
var a=5, b=10, c=4;	
if(a%2==0) // false document.write("a değişkeni çift sayıdır");// çalışmaz if(b%2==0) // true document.write("b değişkeni çift sayıdır");// çalışır	

```

if(c%2==0) // true
    document.write("c değişkeni çift sayıdır");// çalışır
ÇIKTI
b değişkeni çift sayıdır
c değişkeni çift sayıdır

```

If komutundaki *komutlar* eğer istenirse, *if* satırının devamına da yazılabilir. Ancak bu *okunabilirliği* zorlaştırmalıdır, o nedenle yukarıdaki örneklerdeki gibi komutların if satırından sonra *alt satırda* yazılması ve satır başından *girinti* yapılarak hizalanması daha doğru olacaktır.

Tablo 10.30. Okunabilirlik Açısından Karmaşık Ve Düzenli Yazım Biçimleri

Karmaşık yazım şekli	Önerilen hizalı yazım şekli
<pre> var a=5, b=10, c=4; if(a % 2 == 0) b = 2 * a + c; else b = a / 2 +c; </pre>	<pre> var a=5, b=10, c=4; if(a % 2 == 0) b = 2 * a + c; else b = a / 2 +c; </pre>

Switch Komutu

Program akışının bir koşula göre değişmesi için kullanılan bir diğer komut ise *switch* komutudur. Komutun yazılış şekli aşağıdaki şekildeki gibidir.



Her case yanına tek bir değer yazılabilir.

Tablo 10.31. Switch Komutu Örneği

Javascript Komutları
<pre> Var gun=3; var gunAdi; switch(gun) { case 1: gunAdi = "Pazartesi"; break; case 2: gunAdi = "Salı"; break; case 3: gunAdi = "Çarşamba"; break; case 4: gunAdi = "Perşembe"; break; case 5: gunAdi = "Cuma"; break; case 6: gunAdi = "Cumartesi"; break; case 7: gunAdi = "Pazar"; break; default: gunAdi = "belirsiz"; break; } </pre>

```
document.write("Gün adı = " + gunAdı);
```

ÇIKTI

Gün adı = Çarşamba

Yukarıdaki **switch** komutunda, parantez içerisindeki değişkenin (**gun**) değeri (3) ile yukarıdan aşağıya doğru **case** ifadelerinin sağındaki değerler (1, 2,... vb.) karşılaştırılır. Eğer bu değer ile **case** değeri eşit ise, o **case** ifadesinden sonraki komutlar çalıştırılır. Case ifadesinden sonra, **switch** parantezi içerisindeki değişkenin değeri olabilecek değerler yazılmalıdır. Bunun için *istenilen sayıda case* ifadesi ve değeri yazılabılır.

Case ifadesinden sonra yapılacak komutların en sonuna ve bir sonraki **case**'den önce **break** komutu eklenmelidir. Break komutu ile devamındaki **case** komutlarının kontrol edilmesi ve çalıştırılması engellenir ve switch komutu sonlandırılır. Eğer **break** komutu yazılmaz ise değeri eşleşen **case** ifadesinin komutları çalıştırılır *ve devamındaki case* ifadesinin komutları da, *değeri eşleşmese bile*, çalıştırılır.

En sondaki **default** ifadesini yazmak şart değildir, ancak yapılacak ise *bir defa* yer almmalıdır. **Default** ifadesi, var olan **case** değerlerinden hiç biri ile eşleşme olmuyorsa çalıştırılmak istenilen komutlar için eklenir. Genellikle **default** ifadesi tüm **case** ifadelerinden sonra yazılır ancak bu şart değildir. Eğer araya yapılacak ise **break** komutu ile sonlandırılması gereklidir.



Switch komutu içerisindeki break komutu yazılmazsa söz dizim hatası olmaz, ancak mantık hatası olabilir.

Tablo 10.32. Switch Komutundaki Default İfadesi

Javascript Komutları

```
var gun=10;      // eşleşme olmazsa, default kısmı çalışır
var gunAdı;
switch(gun) {
    case 1:
        gunAdı = "Pazartesi";
        break;
    case 2:
        gunAdı = "Salı";
        break;
    case 3:
        gunAdı = "Çarşamba";
        break;
    case 4:
        gunAdı = "Perşembe";
        break;
    case 5:
        gunAdı = "Cuma";
        break;
    case 6:
        gunAdı = "Cumartesi";
        break;
    case 7:
        gunAdı = "Pazar";
        break;
    default:
        gunAdı = "belirsiz";
        break;
}
```

```

}
document.write("Gün adı = " + gunAdı);

```

ÇIKTI

Gün adı = belirsiz

Eğer farklı değerler için aynı komutlar çalıştırılmak istenirse, **her değer** için bir **case** yazılır ancak **break** yazılmadan aşağıdaki gibi boş bırakılır.

Tablo 10.33. Birden Fazla Değerin Eşleşmesinin Sağlanması

Javascript Komutları

```

var gun=3;
var gunAdı;
switch(gun) {
    case 1:
    case 2:
    case 3:
    case 4:
    case 5:
        gunAdı = "hafta içi";
        break;
    case 6:
    case 7:
        gunAdı = "hafta sonu";
        break;
    default:
        gunAdı = "belirsiz";
}
document.write("Gün adı = " + gunAdı);

```

ÇIKTI

Gün adı = hafta içi

Koşul Operatörü (?:)

Bir koşula bağlı olarak, iki farklı değerden birini seçen operatör, koşul operatörüdür (?:). Kullanımı aşağıdaki gibidir.

- Değişken = koşul ? değer1 : değer2;

Öncelikle **koşulun** sonucuna bakılır, sonuç **true** ise **değer1** olarak ifade edilen değer **Değişken'e** aktarılır. Eğer sonuç **false** ise **değer2** olarak ifade edilen değer **Değişken'e** aktarılır.

Tablo 10.34. Koşul Operatörünün Örneği

Javascript Komutları

```

var a = 6, sonuc;
sonuc = a % 2 == 0 ? "çift" : "tek";
document.write("a değişkeni " + sonuc + " sayıdır");

```

ÇIKTI

a değişkeni çift sayıdır



? : operatörünün
sonundaki noktalı virgül
(;) unutulmamalıdır.

Koşul operatörü ile yapılan işlem, **if..else** komutu ile yapılabilir, ancak daha kısa olduğu için tercih edilebilir.

Tablo 10.35. Koşul Operatörü Ve If...Else Komutunun Benzerliği

Koşul Operatörü ile Çözüm	
<pre>var a = 6, sonuc; sonuc = a % 2 == 0 ? "çift" : "tek"; document.write("a değişkeni " + sonuc + " sayıdır");</pre>	
If Else Komutu ile Çözüm	
var a = 6, sonuc;	
if(a % 2 == 0)	
sonuc = "çift";	
else	
sonuc = "tek";	
document.write("a değişkeni " + sonuc + " sayıdır");	



Özet

• Giriş

- Javascript, dil özellikleri bakımında C/C++ programlama dillerine benzer. Komutlar noktalı virgül (;) ile biter, bloklar küme parantezleri { } ile açılır ve kapanır. Eğer istenirse tüm komutlar aynı satır yan yana yazılabilir. Ancak kodların anlaşılır olması için komutların alt alta, her satırda tek komut olacak şekilde yazılması ve satır başı girintilerinin uygulanması tavsiye edilir. Bu sayede kodlama aşamasında birçok olası hata fark edilebilir. Bu üitede, Javascript'te değişkenler, operatörler ve koşul ifadelerini ele alacağız.

• JAVASCRIPT VE HTML ETİKETLERİ İLİŞKİSİ

- Javascript dili ile HTML sayfalarındaki tüm etiketlerin özellikleri ve/veya içerikleri değiştirilebilir. Bunun için Javascript komutları içerisinde ilgili etikete document.getElementById() komutu ile erişilebilir. Komuttaki parantezler içeresine yazılacak isim (id), etikete verilmiş "id" ismi olmalıdır.

- Üç komut ile farklı yerlere çıktı yazdırma işlemi yapılabilir.

- alert();

- document.write();

- console.log();

- Açıklama satırları // sembollerile başlar. Eğer birden fazla satır açıklama satırı yapılacak ise satırlar /* ve */ ifadeleri arasına yazılmalıdır.

• Değişkenler

- Javascript'te değişkenler mutlaka önceden tanımlanmalıdır. Tanımlama için var kelimesi kullanılır.

- Değişken isimlerinde bir önceki üitede belirtilen isimlendirme kuralları geçerlidir.

• Değişken Türleri ve Değerleri

- Değişkenlere tanılandıkları yerde ilk değer vermek tavsiye edilir. Bu sayede hem türü belirlenmiş olur hem de beklenmedik sonuçlar ortaya çıkmaz.

Değişkenleri türü ilk alacakları değerin türü ile belirlenir. Javascript'teki veri türleri sayı (number), metin (string), belirsiz (undefined), nesne (object) ve ikilik (boolean).

- Değişkenlerin veya açık değerlerin türlerini kod içerisinde anlamak için typeof() komutu kullanılır.

- Değişkenler henüz ilk değerini almadıklarında tür ve değer olarak undefined şeklinde tanımlanır. Herhangi bir değişken açık biçimde undefined değerine atanarak hem içerisindeki değer silinmiş olur, hem de türü undefined olarak sıfırlamış olur. Bu değişkene daha sonra farklı türde bir değer verilerek türü değiştirilebilir.

- Null değeri ise nesnelerin (object türündeki değişkenlerin) değerini yok etmek için kullanılır. Null değerini alan nesnenin içindeki değer silinirken, türü yine object olarak kalır. Eğer nesnenin hem değerini hem de türünü sıfırlamak istenirse undefined değerine atanmalıdır.



Özet (devamı)

- **Operatörler**

- Değişkenlerin değerlerini değiştirmeye veya değerlerini kontrol etmeye yarayan komutlara operatör denir. Operatörler genellikler sembollerden oluşur. Yaptıkları işler bakımından operatörler 3 sınıfa ayrılabilir:

- **Aritmetik Operatörleri**

- Sayısal türdeki değişkenler ve / veya değerler ile yapılacak aritmetiksel işlemleri yapan operatörlerdir (+, -, *, /, %, ++, -- vb).

- **Karşılaştırma Operatörleri**

- Değişken ve/veya değerleri birbirine göre kıyaslayan ve sonuç olanı doğru (true) ya da yanlış (false) değerlerini üreten operatörlerdir (<, >, ==, <= >:, != vb).

- **Mantık Operatörleri**

- Birden fazla karşılaştırma işleminin sonrasında tek bir doğru ya da yanlış kararı verebilmek için kullanılan operatörlerdir (&&, ||, !).

- **Operatörlerde İşlem Önceliği ve Yönü**

- Aynı satırda yan yana yazılan operatörler beraberce sırayla çalışırlar. Bu çalışma sırasında işlem önceliği denir. Eğer işlem önceliği aynı olan operatörler varsa o zamanda işlem yönüne göre (soldan sağa ya da tam tersi) işlem yapılır.

- İşlem önceliğini değiştirerek belirli işlemlere öncelik verilmek istenirse o işlem parantez içerisinde yazılmalıdır.

- **Koşul Komutları**

- Javascript'te iki adet koşul komutu vardır:

- **if Komutu**

- Bir koşula göre kimi komutların çalışmasını veya çalışmamasını sağlar. Operatörler ile yapılan işlemlerin sonucundaki doğru (true) ya da yanlış (false) sonucuna göre akış yönlendirilir.

- **Switch Komutu**

- Bir değişkenin olası değerlerine göre akış yönlendirilir. Case ifadesi ile olası değerler ve olası komutlar yazılır. Eğer herhangi bir case değeri ile eşleşme olursa, devamındaki komutlar çalıştırılır.

- **Koşul Operatörü**

- Koşul operatörü ?: sembolleridir.

DEĞERLENDİRME SORULARI

1. Aşağıdakilerden hangisi id'si "aciklama" olan bir paragraf etiketinin içerisindeki yazıyı değiştirir?
 - a) `document.getElementById.aciklama.innetText = "dikkat";`
 - b) `document.getElement(aciklama).innetText = "dikkat";`
 - c) `document.getElementById("aciklama").innetText = "dikkat";`
 - d) `document.aciklama.innetText = "dikkat";`
 - e) `document.getElement("aciklama").innetText = "dikkat";`
2. Aşağıdaki komutlardan hangisi web tarayıcısına yeni pencere açarak çıktı verir?
 - a) `document.getElementById.body.Text("dikkat");`
 - b) `alert("dikkat");`
 - c) `document.alert("dikkat");`
 - d) `alert.open("dikkat");`
 - e) `concole.log("dikkat");`
3. Aşağıdaki sembollerden hangisi açıklama satırlarının başına konulmalıdır?
 - a) *
 - b) /*
 - c) **
 - d) !/
 - e) //
4. Aşağıdaki değişkenlerden hangisi boolean türündedir?
 - a) `var a=1;`
 - b) `var a="1";`
 - c) `var a=null;`
 - d) `var a=true;`
 - e) `var a=undefined;`
5. Aşağıdaki tanımlamadan sonra, `typeof(a)` ifadesinin çıktısı nedir?

```
var a;
```

 - a) number
 - b) string
 - c) object
 - d) boolean
 - e) undefined

6. Aşağıdaki tanımlamadan sonra, a değişkenin değeri nedir?

```
var a="20 + 40";  
a) 2040  
b) "2040"  
c) "20 + 40"  
d) 60  
e) "60"
```

7. Aşağıdaki işlemin sonunda sonuc değişkeninin değeri nedir?

```
var sonuc = 20 / 5 + 5 % 2;  
a) 5  
b) 1  
c) 0  
d) 4  
e) 3.82
```

8. Aşağıdaki işlemin sonunda sonuc değişkeninin değeri nedir?

```
var a=3, b=4, c=5;  
var sonuc = a > b || c % b == 0;  
a) true  
b) false  
c) 0  
d) 8  
e) 11
```

9. Aşağıdaki işlemin sonucunda a değişkeninin değeri ne olur?

```
var a=10, b=5;  
a += b++;  
a) 15  
b) 5  
c) 6  
d) 16  
e) 105
```

10. Aşağıdaki işlemin sonucunda t değişkeninin değeri ne olur?

```
var t=1, y=5;  
if(t != y % 2)  
    t *= 5;  
else  
    t += y;  
a) false  
b) true  
c) 6  
d) 5  
e) 15
```

Cevap Anahtarı

1.c, 2.b, 3.e, 4.d, 5.e, 6.c, 7.a, 8.b, 9.a 10.c

YARARLANILAN KAYNAKLAR

JavaScript. 18 Temmuz 2019 tarihinde <https://tr.wikibooks.org/wiki/JavaScript> adresinden erişildi.

JavaScript Operator Precedence. 19 Temmuz 2019 tarihinde <https://www.dummies.com/web-design-development/javascript-operator-precedence/> adresinden erişildi.

Javascript Tutorial. 18 Temmuz 2019 tarihinde <https://www.w3schools.com/js/> adresinden erişildi.

Javascript Dersleri. 7 Temmuz 2019 tarihinde <http://javascript.sitesi.web.tr/> adresinden erişildi.

JAVASCRIPT'TE İLERİ KONULAR

İÇİNDEKİLER



- Metin İşlemleri
- Döngü Komutları
- Diziler
- Fonksiyonlar
- Olaylar
- DOM ile Programlama
- Window Nesnesi

HEDEFLER

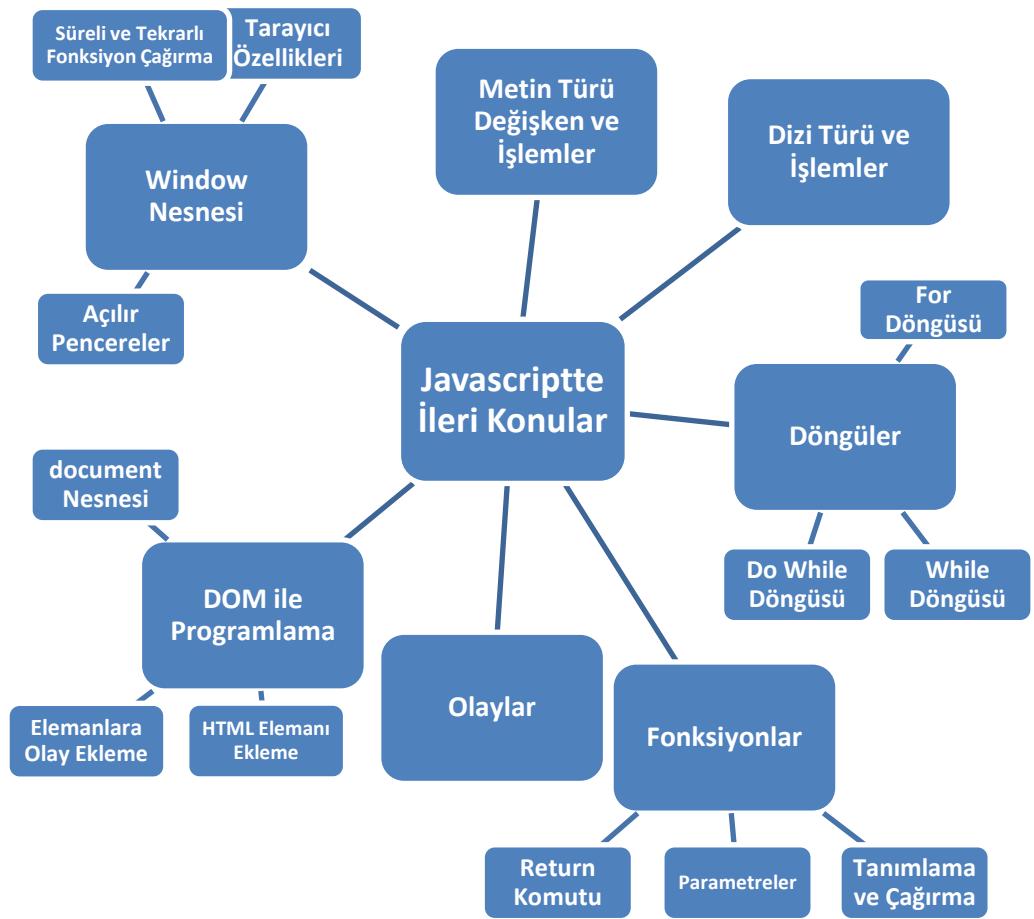
- Bu üniteyi çalıştıktan sonra;
 - Javascript'te metinler üzerinde işlem yapabilecek,
 - Dizileri ve döngüleri programlarınızda kullanabilecek,
 - Farklı türde fonksiyon tanımlayabilecek ve kullanabilecek,
 - DOM ile safta elemanlarına erişebilecek,
 - Window nesnesi ile tarayıcı üzerinde ileri işlemler yapabileceksiniz.



Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET
PROGRAMCILIĞI II
Dr. Öğr. Üyesi Halil
ERSOY

ÜNİTE
11



GİRİŞ

Önceki iki üitede Javascript'in temelleri olabilecek konular işlenmişti. Bu üitede ise Javascript dili ile yapılabilecek ileri düzey işlemler için gerekli olacak diğer konular ele alınacaktır.

Değişken türlerinde kısaca bahsedilen *metin (string)* türü üzerinde yapılabilecek birçok işlem mevcuttur. Metinler ile yapılabilecek işlemler iki açıdan önemlidir. İlk metin türünde girilmiş değerler üzerinde işlem yapmak, ikincisi HTML etiketlerini metin türünde düzenlemek. Her iki durum aşağıdaki bölümde detaylıca ele alınacaktır.

Programın akışını kontrol eden koşul yapıları önceki üitede ele alınmıştır. Bunlara ek aynı işlemlerin birden fazla defa yapılması istendiğinde komutların *döngüler* içerisinde yazılması konusu, bu üitede gösterilecektir. For, while ve do döngüleri bu amaçla üitede ele alınacaktır.

Yine veri türü olarak kısaca önceki üitede ifade edilen *diziler*, bu üitede detaylıca anlatılacaktır. Dizilerin döngüler ile bir arada kullanılması çok yaygın bir uygulamadır. Bu nedenle her iki konu birbirini destekleyici niteliktedir.

Yazılan komutların satır satısı arttıkça, programların anlaşılması ve olası hataların ayıklanması güçleşir. Ayrıca bir işi yapan komutların kümeler hâlinde tekrar tekrar kullanılması durumunda, satır sayıları anlamsızca artar. Bu karmaşıklığı ve gereksiz tekrarlamayı engellemek için komutlarımıza *fonksiyonlar* hâlinde yazmaya çalışacağız. Fonksiyonlar konusu tipki diğer diller gibi Javascript için de önemli bir konudur ve bu üitede ele alınacaktır.

Javascript'in HTML etiketleri üzerindeki etkisini göstermek için *DOM* (Document Object Model) modelinin bilinmesi gereklidir. Bu model çerçevesinde hangi etikete ne şekilde erişebileceğimiz belirlenir. Aynı zamanda bu modele göre HTML etiketlerinin veya kullanıcı hareketlerinin tetiklediği *olaylar* da bu üitede ele alınacaktır.

Son olarak DOM içerisindeki *window* nesnesinin tarayıcı özellikleri, tarayıcı geçmişi ve açılı pencereler gibi metotları anlatılmıştır.

METİN TÜRÜ VE İŞLEMLERİ



Metin türündeki değerleri çift tırnak ya da tek tırnak ile sınırlayabilirsiniz.

Değişken ve veri türü olarak metin (string) türünü daha önceki üitede görmüştük. Aşağıdaki örnekte tüm değişkenlerin türü metindir. Dikkat edilirse hepsinin çift ya da tek tırnak içerisinde olduğu görülmektedir.

Tablo 11.1. Metin Türü Değişken Ve Değerler

Javascript Komutları

```
<script>
var a="Merhaba dünya";
var b='Merhaba dünya';
var c="3.1417";
var d="Ankara'da hava sıcak";
```

```
var e=<p>Hikaye başlıyor...</p>";
</script>
```

Metin değerlerini sınırlamak için başına ve sonuna tek tırnak (ya da kesme) veya çift tırnak karakterleri kullanılır. Ancak hangisi ile başlanırsa yine aynısı ile sonlandırılmalıdır.

Eğer metin içinde simbol olarak tek tırnak ya da çift tırnağa ihtiyaç varsa önüne ters bölü (\) simbolü eklenerek metin içerisinde yer verilebilir.



Boşluk, noktalama işareti, matematiksel semboller gibi tüm karakterler metin değişkenlerinde yer kaplar.

Örnek

```
•var d="Ankara'da hava sıcak<br>";
•var e='Ankara\'da hava sıcak<br>';
•var f="Ankara \"sıcaklık\" artıyor<br>";
•document.write(d); // Çıktı : Ankara'da hava sıcak
•document.write(e); // Çıktı : Ankara'da hava sıcak
•document.write(f); // Çıktı : Ankara "sıcaklık" artıyor
```

Metin türündeki değişkenler, aktarıldığı değişken içerisinde aslında sıraya dizilmiş karakter kümeleridir. Dolayısı ile her bir karaktere sırasını belirten bir sayı ile aşağıdaki gibi erişilebilir.

```
var a="Merhaba dünya";
```

M	e	r	h	a	b	a		d	ü	n	y	a
a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]	a[10]	a[11]	a[12]

Şekil 11.1. Metin karakterlerinin indis numaraları

- document.write(a[0]); // →'M'
- document.write(a[12]); // →'a'

Sıra (indis) *numarası 0'dan* başlar, bu nedenle metindeki ile karakterin sıra numarası 0'dır. Son karakterin numarası ile *karakter sayısının bir eksigidir*. Bu sıra numarasına bundan sonra indis denilecektir.

Javascript'te metin üzerinde yapılacak işlemlerde kullanılacak hazır fonksiyonlar aşağıda anlatılmıştır. Bu fonksiyonlarda hangi karakter veya karakterler üzerinde işlem yapılacak ilgili karakterlerin *indis numaraları* üzerinden anlatılmıştır.

Metin Uzunluğu

Bir metindeki karakter sayısına o metnin karakter uzunluğu denir. Karakter uzunluğu görünen veya görünmeyen tüm karakterlerin toplamıdır. Bir metnin uzunluğunu bulmak için metin türündeki değişkenin *length* özelliği kullanılır.

Tablo 11.2. Length Özelliği İle Metin Uzunluğu Bulma**Javascript Komutları**

```
var a="Merhaba dünya";
document.write(a.length); → 13
var b="Ankara'da";
document.write(b.length); → 9
var c="1350 TL";
document.write(c.length); → 7
```

Length özelliği metni oluşturan tüm karakterlerin adedini verir. Buna göre metnin *ilk karakterinin indis 0* iken, *son karakterin indis a.length-1*'dir.

Büyük ve Küçük Harf Kontrolü ve Çevirme

Metindeki karakterlerin büyük harfe ya da küçük harfe çevrilmesi için *toUpperCase()* ve *toLowerCase()* özellikleri kullanılır.

Tablo 11.3. Büyük Ve Küçük Harfe Çevirme

“toUpperCase” ve
“toLowerCase”
fonksiyonları hem bir
metin hem de bir
karakter için çalışır.

Javascript Komutları

```
var a="Merhaba dünya";
a=a.toUpperCase(); → MERHABA DÜNYA
a=a.toLowerCase(); → merhaba dünya
```

Yukarıdaki iki fonksiyonda dikkat edilmesi gereken, fonksiyonların çağrıldıkları metin içerisindeki değerleri değiştirip geriye değişmiş hâlini döndürdükleridir. Aşağıdaki örneği dikkatli inceleyin.

Tablo 11.4. Touppercase Fonksiyonu**Javascript Komutları**

```
var a="Merhaba dünya";
document.write(a); → Merhaba dünya
document.write(a.toUpperCase()); → MERHABA DÜNYA
document.write(a); → Merhaba dünya
a=a.toUpperCase();
document.write(a); → MERHABA DÜNYA
```

ToUpperCase ve *toLowerCase* fonksiyonları metin içerisindeki rakam veya noktalama işaretlerinde işlem yapmaz.

Eğer metindeki bir karakterin büyük ya da küçük olduğunu sorgulamak isterseniz aşağıdaki örnekte gösterildiği üzere karakterin, kendisinin büyük ya da küçük hâline eşit olup olmadığı kontrol edilebilir.

Tablo 11.5. Küçük Ya Da Büyük Harf Kontrolü Örneği**Javascript Komutları**

```
var a="Merhaba dünya";
if(a[0] == a[0].toUpperCase())
    document.write("ilk karakter büyük")
else
    document.write("ilk karakter küçük")
```

Çıktı:

İlk karakter büyük

Metin İçinde Arama Yapma

Metin içerisinde karakter ya da başka metin aramak için *indexOf()* ve *lastIndexOf()* fonksiyonları kullanılır. Bu fonksiyonlar aradıkları ifadenin metin içerisindeki, varsa, *indis numarasını* geri döndürürler.



IndexOf ve lastIndexOf metotları sadece ilk buldukları karakterin yerini gösterir, devamını kontrol etmez.

IndexOf() baştan, *lastIndexOf()* ise sonda aramaya başlar. Eğer ifade bulunmazsa -1 değerini döndürürler. Arama yine büyük ve küçük harfe duyarlıdır.

Tablo 11.6. Indexof Ve Lastindexof Fonksiyonları**Javascript Komutları**

```
var a="Merhaba dünya";
var konum1=a.indexOf("a");    → 4
var konum2=a.lastIndexOf("a"); → 12
var konum3=a.indexOf("A");    → -1
```

Metnin Karakterleri Üzerinde İşlem Yapma

Metnin karakterlerine, karakter dizileri gibi *indis numarası* ile erişilebilir. Ancak bu metnin bir dizi olduğu anlamına gelmez. İndis numarası ile karakterler ulaşabiliriz ancak onları bu yolla *değiştiremeyiz*.

Tablo 11.7. Karakterlere İndisle Erişme**Javascript Komutları**

```
var a="Merhaba dünya";
a[0]= "T";      // Hata vermez ancak değişiklik yapmaz!
document.write(a); // Çıktı: Merhaba dünya
```



Karakter dizisi ile metin tamamen aynı türde değişken değildir, ancak birbirine dönüştürülebilir.

Eğer karakterler üzerinde işlem yapmak istersek o *zaman metni önce karakter dizisine dönüştürmek* gerekir. Dizi konusu ileride anlatılacaktır ancak metinlerin diziye dönüştürülmesini aşağıdaki örnek ile burada görelim.

Tablo 11.8. Karakterleri Diziye Çevirme Ve Birleştirme**Javascript Komutları**

```
var a="Merhaba dünya";
var k=a.split(""); // karakterlere göre parçalama
document.write(a);

k[0]="T";        // İlk karakteri T yapma
document.write(k);
a=k.join("");    // Karakterleri birleştirme ve string yapma
document.write(a);
```

Çıktı:

Merhaba dünya T,e,r,h,a,b,a, ,d,ü,n,y,a Terhaba dünya

DÖNGÜLER

Programlama sürecinde bir ya da birden çok komutun, bir kez yazıldığı hâlde birden çok kez tekrar çalışmasını sağlayan komutlara döngüler denir. Döngüler tıpkı koşul komutları gibi programlama dillerinin temel komutlarıdır ve programların daha az sayıda satır ile yazılabilmelerini sağlar.

Javascript'te üç farklı komut döngü komutu vardır:

- for
- while
- do while

Bu üç döngü aşağıda ayrı ayrı ele alınmıştır. Öte taraftan bu üç döngünün de ek değişkenler ile birbiri yerine kullanılabilecekleri unutulmamalıdır.

For Döngüsü

Tekrarlanma sayısı belirli olan durumlarda **for** döngüsü kullanmak tavsiye edilir. For döngüsünün yazım şekli aşağıdaki gibidir:

Tablo 11.9. For Döngüsü Yapısı

For Döngüsü Yapısı ve Tanımlar	
	<pre>for(komut1; koşulKomutu; komut3) { tekrarlanması istenilen komutlar; tekrarlanması istenilen komutlar; tekrarlanması istenilen komutlar; }</pre>
komut1; örneğin i=0; yada sayac=50;	Bu komut ilk sırada ve bir defa çalıştırılır. Genellikle döngüde kullanılacak sayaç değişkeninin ilk değerini vermek için kullanılır. Eğer istenirse birden fazla değişkene, aralarına virgül konularak ilk değer verilebilir. Sonunda noktalı virgül yer almmalıdır.
koşulKomutu; örneğin i<100; ya da sayac%5==0;	Komut1'den sonra çalıştırılır, sonucun true (doğru) olması durumunda döngü içerisindeki "tekrarlanması istenilen komutlar" bloku çalıştırılır. Eğer koşulKomutu değeri false (yanlış) olursa for döngüsü sona erer ve program akışı for döngüsü bloku dışından devam eder.
komut3; örneğin i++; sayac--;	Blok çalışıktan sonra komut3 çalıştırılır. Genellikle, sayaç olarak kullanılan değişkenin artma ya da azaltma komutu verilir. Bu komuttan sonra akış yine koşulKomutuna geçer ve koşulun sonucuna göre ya blok çalıştırılır ya da döngü sona erer.



For döngüsü dizilerle birlikte sıkılıkla kullanılır.

Aşağıdaki örnekte web sayfasında bir tabloya 5 satır eklenmiş olur. Bu örnekteki **sayac** değişkeni ilk değeri olan 1 ile döngü başlar. Ardında sayac++ komutu ile değeri 1 artarak sırasıyla 2, 3, 4 ve 5 olur ve for döngüsü altındaki

`document.write()` komutu 5 kez çalışır. Artış komutu `sayac` değişkeninin değerini 6 yapar, ancak 6 değeri koşulda `false` sonucunu getirdiği için döngü sona erer.

Tablo 11.10. For Döngüsü ile Tabloya Satır Ekleme

Javascript Komutları								
<table border="1" width="200">								
ÇIKTI								
<script> var sayac; for(sayac=1; sayac<=5; sayac++) document.write("<tr><td>Merhaba dünya</td></tr>"); </script>	<table border="1"> <tr><td>Merhaba dünya</td></tr> <tr><td>Merhaba dünya</td></tr> <tr><td>Merhaba dünya</td></tr> <tr><td>Merhaba dünya</td></tr> <tr><td>Merhaba dünya</td></tr> </table>	Merhaba dünya a						
Merhaba dünya								
Merhaba dünya								
Merhaba dünya								
Merhaba dünya								

Döngü kullanımında yapılan iki tür hata vardır: Döngünün hiç başlamaması ya da döngünün hiç bitmemesi, yani sonsuz döngü durumu. Her iki durum da çoğu zaman programcının mantık hatasından kaynaklanır. Bu iki hata tüm döngülerde yapılabilir. Aşağıdaki iki örnekten ilkinde `for` döngüsünün hiç başlamadığı, ikincisinde ise döngünün sona eremediği görülebilir.



Web sayfalarında
oluşacak **sonsuz**
döngüler, o noktadan
sonra sayfanın
yüklenmesini
engelleyebilir ya da
sayfayı kullanılamaz
hâle sokabilir.



Örnek

- Hatalı döngü örneği: Döngü hiç başlamaz!

```
var sayac;  
for(sayac=1; sayac>=5; sayac++)  
    document.write("Merhaba");
```

- Hatalı döngü örneği: Döngü başlar ancak hiç bitmez!

```
var sayac;  
for(sayac=1; sayac<=5; sayac--)  
    document.write("Merhaba");
```

`For` döngüsünün yazım şeklindeki 3 kısımdan (komut1, koşulKomutu ve komut3) herhangi biri istenirse yazılmayabilir. Bu durumda eksik kısmın işlevini yapacak ek komutlar döngü öncesinde ya da döngü içerisinde yazılmalıdır. Aşağıdaki örneklerde komut1 ve komut3'ün olmadığı for döngüleri görülebilir. Bu komutlar geçerli olarak çalışır ancak profesyonel olarak anlaşılmasının zor olduğu için önerilmez.

Tablo 11.11. For Döngüsü Örnekleri

Javascript Komutları	ÇIKTI
<pre>var sayac=1; for(; sayac<=5; sayac++) document.write(sayac + " . satır
")</pre>	1 . satır 2 . satır 3 . satır 4 . satır 5 . satır
<pre>var sayac=1; for(; sayac<=5;){ document.write(sayac+ " . satır
"); sayac++; }</pre>	1 . satır 2 . satır 3 . satır 4 . satır 5 . satır



For döngüsünde bazı komutlar parantez içine yazılmasa bile noktalı virgül karakterleri (;) mutlaka yazılmalıdır.

While Döngüsü

Tekrar etmesini istediğimiz komutların, belirli bir koşul sağlandığı sürece tekrar çalıştırılmasını sağlayan döngü **while** döngüsüdür.

Tablo 11.12. While Döngüsü Yapısı

While Döngüsü Yapısı	
while(koşulKomutu) { tekrarlanması istenilen komutlar; tekrarlanması istenilen komutlar; tekrarlanması istenilen komutlar; }	koşulKomutu Bu koşul true veya false üreten bir ifade olmalıdır. İfade true (doğru) değerini ürettiğい sürece döngü içerisindeki komutlar çalıştırılır.

While döngüsünde, daha önce bahsedilen iki hata riski daha fazladır. Döngünün hiç başlamaması veya döngünün hiç bitmemesi için koşul dikkatli yazılmalı ve döngü bloku içerisinde bu koşulu er ya da geç false değerine dönüştürecek işlemler unutulmamalıdır.

Tablo 11.13. While Döngüsü Örneği

Javascript Komutları	
<pre><table border="1" width="200"> <script> var sayac=1; // önemli while(sayac<=5){ document.write("<tr><td>Merhaba dünya</td></tr>"); sayac++; // önemli } </script> </table></pre>	ÇIKTI

Merhaba dünya
ası ve bitebilmesi için dikkatlice yazılmalıdır.

Do While Döngüsü

Tekrar edilecek işlemlerin en az bir defa çalıştırılması ve devamında bir koşula göre tekrar çalıştırılması gerekirse, kullanılması gereken döngü **do while** olabilir. **Do while** döngüsünün yapısı aşağıdaki gibidir.

Tablo 11.14. Do While Döngüsü

Do while döngüsü yapısı	
do{ tekrarlanması istenilen komutlar; tekrarlanması istenilen komutlar; tekrarlanması istenilen komutlar; }while(koşulKomutu);	Once blok içerisindeki komutlar bir kez çalıştırılır. Ardından true veya false üreten koşulKomutu kontrol edilir. Sonuç true (doğru) değerini ürettiği sürece döngü içerisindeki komutlar çalıştırılır.
koşulKomutu	


While ve **do..while** döngüleri birbirine dönüştürülebilir.

Do while döngüsünde koşulun blok sonunda olmasından anlaşılacağı üzere, döngü içerisindeki komutlar koşuldan **once** bir defa çalıştırılır. Ardından koşul kontrol edilir. Diğer döngülerdeki gibi döngünün sonsuz döngüye dönüşmemesi veya tekrar etmesini engelleyecek koşulların yazılmasına özen gösterilmeliidir.

DİZİLER

Aynı türde birden fazla değeri ayrı ayrı tutabilen değişkenlere **dizi** (**array**) denir. Kimi programlama dillerinde diziler bir değişken türü, kimilerinde ise veri tutan nesne türü olarak tanımlanmıştır. Javascript'te bu yapılar tür olarak **nesne** (**object**) türündedir.

Birçok dilden farklı olarak Javascript'te dizilerin her bir elemanı farklı türlerde değer tutabılır. Ancak bu ünitede sadece tüm değerleri aynı türde değer tutan diziler ele alınmıştır.

Dizi Tanımlama

Diziler aşağıdaki gibi tanımlanabilir.

Tablo 11.15. Dizi Tanımlama Örnekleri

Javascript Komutları
<pre>var dizi1=[2,4,6,8,10]; document.write("dizi1 elemanları=" + dizi1 + "
");</pre>
<pre>var dizi2=["TL", "USD", "EU"]; document.write("dizi2 elemanları=" + dizi2 + "
");</pre>
<pre>var dizi3=new Array(5,10,15,20); document.write("dizi3 elemanları=" + dizi3 + "
");</pre>
ÇIKTI
dizi1 elemanları=2,4,6,8,10 dizi2 elemanları=TL,USD,EU dizi3 elemanları=5,10,15,20



Metin türündeki değişkenler dizi gibi indisli bir yapıya sahiptir ancak tam anlamıyla dizi değildir.

Yukarıdaki örnekten anlaşıllacağı üzere;

- Diziler **farklı** türlerde değerleri saklayabilir.
- Dizilere ilk değerler, **köşeli parantez** içerisinde ve aralarına virgül yazılarak verilebilir.
- Dizilere ilk değerler, **new Array()** komutunun parantezleri içerisinde aralarına virgül yazılarak verilebilir.
- **document.write()** komutu dizinin tüm elemanlarını, ekrana aralarına **virgül** koyarak yazabilir.

Dizileri yaratırken **new Array()** komutunun kullanılmasına gerek yoktur. Yukarıdaki örnekte dizi1 ve dizi3 aynı özelliklere sahip dizilerdir. **New Array()** komutunun kullanılmasının tek avantajı, eleman sayısı belli ancak değerleri henüz belirlenmemiş dizi yaratabilmektir.



Örnek

- var d1=[]; // 0 elemanlı bir dizi
- var d2=new Array(); // 0 elemanlı bir dizi
- var d3=new Array(5); // 5 elemanlı bir dizi

Öte taraftan **new Array()** komutundaki parantezlerin içerişine yazılan değerlere dikkat etmek gereklidir. Eğer tek bir tam sayı yazılırsa, o zaman bu sayı dizinin **eleman sayısı** olarak algılanır. Eğer birden fazla değer yazılırsa, bunlar dizinin elemanlarının **ilk değerleri** olarak algılanır.

Tablo 11.16. New Array Komutu Örnekleri

Javascript Komutları ve Çıktılar
var d3=new Array(5); → 5 elemanlı bir dizi
var d3=new Array(5,6,7); → 3 elemanlı bir dizi

Dizilerin Elemanlarına Erişim ve İndis Numaraları

Dizilere atanan değerlere o değerin sırasına göre indis numarası ile teker teker erişilebilir. *İndis numarası* her zaman 0'dan başlar.

Tablo 11.17. Dizi Ve Elemanlarına Erişim Örneği



Dizilerde ilk elemanın indis numarası her zaman sıfırdır.

Javascript Komutları

```
var sembol=["TL", "USD", "EU"];
document.write("sembol elemanları=" + sembol + "<br>");
document.write("sembol[0]="+ sembol[0] + "<br>");
document.write("sembol[1]="+ sembol[1] + "<br>");
document.write("sembol[2]="+ sembol[2] + "<br>");
document.write("<hr>");

sembol[2]="gbp";
document.write("sembol[2]="+ sembol[2] + "<br>");

sembol[2]=sembol[2].toUpperCase(sembol[2]);
document.write("sembol[2]="+ sembol[2] + "<br>");
```

ÇIKTI

```
sembol elemanları=TL,USD,EU
sembol[0]=TL
sembol[1]=USD
sembol[2]=EU

_____
sembol[2]=gbp
sembol[2]=GBP
```

Dizilerin elemanlarının her biri kendi türlerine göre (number/sayı, metin/string, vb.) farklı işlemlere tabi tutulabilir. Yukarıdaki örnekte sembol[2] ifadesi *string* bir değerdir ve *toUpperCase()* fonksiyonuna parametre olarak gönderilebilir.

Dizilerin her bir elemanı *indis numarası* kullanılarak erişilebilir ve değiştirilebilir. Yukarıdaki örnekte sembol[2]= "gbp"; satırında üçüncü değer değiştirilmiştir.

Dizinin Eleman Sayısı

Dizilerin eleman sayısına “dizinin uzunluğu” da denilebilir ve *length* ifadesi ile tamsayı olarak elde edilebilir.

Tablo 11.18. Farklı Dizilerin Length Özelliği



Hiç elemanı olmayan dizilerin eleman sayısı (*length* değeri) 0'dır.

Javascript Komutları

```
var sembol=["TL", "USD", "EU"];document.write("sembol elemanları=" + sembol + "<br>");

document.write("eleman sayısı=" + sembol.length + "<hr>");

var d1=[];
```

```
document.write("d1 elemanları=" + d1 + "<br>");  
document.write("eleman sayısı=" + d1.length + "<hr>");  
  
var d2=new Array(4);  
document.write("d2 elemanları=" + d2 + "<br>");  
document.write("eleman sayısı=" + d2.length + "<hr>");  
  
var d3=[5.0, 3.14, 7.65];  
document.write("d3 elemanları=" + d3 + "<br>");  
document.write("eleman sayısı=" + d3.length + "<hr>");  
ÇIKTI  
sembol elemanları=TL,USD,EU  
eleman sayısı=3  


---

d1 elemanları=  
eleman sayısı=0  


---

d2 elemanları=,,,  
eleman sayısı=4  


---

d3 elemanları=5,3.14,7.65  
eleman sayısı=3
```

Dizilerin Elemanlarına Döngüler ile Erişim

Dizilerin elemanlarının her biri için yapılacak işlemler genellikler aynı olduğu için bu işlemlerin döngüler içerisinde yapılması tercih edilir. Dizilerin eleman sayıları belirli olduğu için (*length* özelliği ile) bu işlemlerde en çok *for* döngüsü tercih edilir.

Tablo 11.19. For Döngüsü İle Elemanlara Erişim



Dizilerin ilk elemanın indisi *sıfır* (0), son elemanın indisi ise *dizi.length-1*'dir .

Javascript Komutları

```
var sayilar=[4,6,2,8, 4,5,12];  
var sayac, toplam = 0;  
var ortalama;  
for(sayac= 0; sayac < sayilar.length; sayac++){  
    toplam += sayilar[sayac];  
}  
ortalama = toplam / sayilar.length;  
document.write("Toplam=" + toplam + "<br>");  
document.write("Eleman sayısı=" + sayilar.length + "<br>");  
document.write("Ortalama=" + ortalama + "<br>");  
ÇIKTI  
Toplam=41  
Eleman sayısı=7  
Ortalama=5.857142857142857
```

Yukarıdaki komutlardan **for** satırında, **koşul ifadesine** dikkat ediniz. Eğer “<” simbolü yerine “<= ” simbolü kullanılırsa, sayaç 7 olacak ve 7. eleman için döngü komutları çalışıp hata oluşacaktır. Bunun sebebi, dizideki son elemanın indisinin 6 olmasıdır. Dizinin sahip olmadığı bir elemana ulaşmak **hata** oluşturur.

Dizi Elemanları İçinde Arama Yapma

Dizilerin elemanları arasında arama yapmak için **indexOf** ya da **lastIndexOf** fonksiyonları kullanılır. **IndexOf** aranan değerin dizinin **başından** itibaren ilk bulunduğu konumun indis numarasını, **lastIndexOf** ise **sondan** itibaren ilk bulunduğu konumun indis numarasını döndürür. Eğer değer **bulunmazsa** -1 değeri döndürülür. Metin türündeki dizilerde arama işlemlerinde **büyük ve küçük harf** fark eder.

Tablo 11.20. Indexof Ve Lastindexof Fonksiyonları ile Arama

Javascript Komutları ve Çıktıları

```
var sayilar=[10,5,2,3,8,1,5];
var sehirler=["İstanbul", "Ankara", "İzmir"];
var sonuc;
sonuc=sayilar.indexOf(5);      → 1 döndürür
sonuc=sayilar.indexOf(4);      → -1 döndürür
sonuc=sayilar.lastIndexOf(5);  → 6 döndür
sonuc=sehirler.lastIndexOf(İzmir); → 2 döndür
sonuc=sehirler.lastIndexOf(izmir); → -1 döndür, i küçük!
```

FONKSİYONLAR



Fonksiyonlar değişken adlandırma kurallarına göre adlandırılır.

Birden fazla komutu gruplayarak tek bir isim ile çağrılabildiğimiz yapılara **fonksiyon** denir. Fonksiyonlar tüm programlama dillerinde benzer şekilde kullanılırlar. Sıklıkla yapılan işlerin tekrarlanması durumunda gereksiz yere yer işgal etmemeleri ve ya farklı değerler için aynı işlemin yapılması gerekiğinde yeniden aynı kodları yazmamak adına bu işler fonksiyon hâline getirilir.

Tablo 11.21. Indexof Ve Lastindexof Fonksiyonları ile Arama

Javascript Komutları ve Çıktıları

```
function topla(a,b){
    return a + b;
}

var sayı1=10, sayı2=30;
var sonuc=topla(sayı1, sayı2);
document.write("Sonuc=" + sonuc); → Sonuc=40

document.write("Sonuc=" + topla(sayı1,sayı2)); → Sonuc=40

document.write("Sonuc=" + topla(5,3)); → Sonuc=8

document.write("Sonuc=" + topla(3.14 , 7.5)); → Sonuc=10.64
```

```
document.write("Sonuc=" + topla("merhaba","dünya"));
→ Sonuc=merhabadünya
```

Yukarıdaki örnekte görüleceği üzere, **topla()** adındaki bir fonksiyon ilk 3 satırda **tanımlanmış**, ardından 5 farklı satırda kullanılmış, yani **çağırılmıştır**. Fonksiyonların isimleri tıpkı değişkenler isimleri gibi amacına uygun biçimde ve isimlendirme kuralları uygulanarak geliştirici tarafından belirlenir.

Fonksiyonlar ancak **tanımlandıktan** sonra **çağırılabilirler**. Tanımlama kısmı ve çağrıma kısmı genellikle aşağıdaki kalıba uygun biçimde yapılır.

Tablo 11.22. Fonksiyon Çağırma Yolları



Bir fonksiyon bir defa tanımlanır ancak birden çok defa çağrırlabilir.

Javascript'te Fonksiyon Tanımlama

```
function fonksiyonAdı (varsayıParametre1, varsayıParametre2, ... )
{
    Fonksiyon içinde çalışması istenen komutlar....  

    return varsayıGeriDönüşDeğeri;
}
```

ÇAĞIRMANIN İKİ YOLU

```
// 1. Geri dönüş değeri varsa:  

var değişken= fonksiyonAdı(argüman1, argüman2...)  

// 2. Geri dönüş değeri yoksa  

fonksiyonAdı(argüman1, argüman2);
```

Parametreler ve Argümanlar

Parametreler aslında değişkenlerdir. Fonksiyonun eğer istenirse çağrıldığı noktadan değişkenler ya da değerler olmasını sağlar. Tanımlama başlığında parantez içerisinde bir ya da birden çok parametre alabilir, eğer parametre istenmezse o zaman parantezlerin içi boş bırakılır.

Tanımlama kısmında eğer **parametre** eklenirse, o zaman o fonksiyon çağrıılma kısmında da aynı sayıda **argüman** ile çağrılmalıdır. Argümanlar değişken ya da açık bir değer olabilir. Aşağıdaki örnekte tanımlamaya, çağrımeye ve açıklamalara dikkat ediniz:

Tablo 11.23. Fonksiyon Tanımlamaları Ve Çağrılmalari

Javascript Komutları

```
// Dört fonksiyon tanımlaması
function paragrafEkle(metin){      // bir parametreli fonk.  

    document.write("<p>" + metin + "</p>");  

}
function baslikEkle(metin){        // bir parametreli fonk.  

    document.write("<h1>" + metin + "</h1>");  

}

function çizgiEkle(){            // parametresiz fonk.  

    document.write("<hr>");
```

```
}
```

```
function ussunuBul(sayı, us){ // iki parametreli ve
    var carpim=1, i; // geri dönüş değerli fonk.
    for(i=0; i<us; i++)
        carpim *= sayı;
    return carpim;
}

// Fonksiyonları çağırılması
var a=2, b=3;
document.write("a^b=" + ussunuBul(a,b) + "<br>"); //iki argüman
document.write("3^3=" + ussunuBul(3,3) + "<br>"); //iki argüman
cizgiEkle(); //argüman yok
baslikEkle("Fonksiyonlar Konusu"); //bir argüman
paragrafEkle("Merhaba, bu bir paragraftır"); //bir argüman
paragrafEkle("Bu da ikinci paragraf olsun"); //bir argüman
cizgiEkle(); //argüman yok
```



Hiç argümanı olmayan fonksiyonları çağrırmak için içi boş parantezler yine de yazılmalıdır.

ÇIKTI:	HTML
<pre>a^b=16 3^3=81</pre> <hr/> <h2>Fonksiyonlar Konusu</h2> <p>Merhaba, bu bir paragraftır</p> <p>Bu da ikinci paragraf olsun</p> <hr/>	<pre>><script>...</script> "a^b=8"
 "3^3=27"
 <hr> <h1>Fonksiyonlar Konusu</h1> <p>Merhaba, bu bir paragraftır</p> <p>Bu da ikinci paragraf olsun</p> <hr></pre>

Parametrelerin değeri fonksiyon içerisinde değişebilir, ancak bu değişiklik temsil ettikleri argümanlara *her zaman yansımaz*. Eğer argümanlar sayısal veya metin türünde değişkenler ise değişiklik yansımaz, ancak parametreler nesne ya da dizi ise değişiklik yansır. Aşağıda buna yönelik iki örnek verilmiştir.

Tablo 11.24. Değişen Parametreler Ve Argümanlara Etkisi

Javascript Komutları

```
function fonksiyon1(x,y){ // Değerleri değiştiren fonksiyon
    var yedek;
    yedek=x;
    x=y;
    y=yedek;
}

function fonksiyon2(z){ // dizinin her bir elemanına 100
    var i; // ekleyen fonksiyon
    for(i=0; i<z.length; i++)
        z[i] += 100;
```

```

}

var a=10, b=20;
var d=[5,10,15,20];

document.write("Fonksiyondan önce a=" + a + " b=" + b + "<br>");
document.write("Fonksiyondan önce d=" + d + "<br>");
fonksiyon1(a,b);
fonksiyon2(d);
document.write("Fonksiyondan sonra a=" + a + " b=" + b + "<br>");
document.write("Fonksiyondan sonra d=" + d + "<br>");
```

ÇIKTI

Fonksiyondan önce a=10 b=20
 Fonksiyondan önce d=5,10,15,20
 Fonksiyondan sonra a=10 b=20 → Değişim yok
 Fonksiyondan sonra d=105,110,115,120 → Değişim var



Return komutu,
fonksiyonu bitiren
komuttur.

Return Komutu

Eğer fonksiyon içerisindeki işlemlerin sonucunda, fonksiyonun çağrııldığı noktaya bir *değer göndermek istenirse*, o zaman fonksiyonun uygun yerine *return* kelimesi ve *yanına* uygun bir değer ya da değişken yazılır. Eğer farklı durumlarda farklı değerler geri döndürülecek ise *birden fazla return yazılabilir*. Fonksiyonların çalışması, eğer varsa, kendi içerisindeki *return* kelimesine ulaşıldığında sona erer. *Return* kelimesinden sonraki komutlar çalıştırılmaz. Öte taraftan eğer *return* kelimesi *yoksa* fonksiyonun tüm komutları çalıştırılır.

Fonksiyonlar, çağrıdıkları anda programın akışını devralır ve kendi içerisindeki komutların çalışması bitene kadar ya da varsa *return* komutuna ulaşana kadar akışı elinde tutar.

Tablo 11.25. Return Kelimesi Örneği

Javascript Komutu	ÇIKTI:
<pre>function kucukOlan(x,y){ if(x<y) return x; return y; } var a=10, b=20, c; c=kucukOlan(a,b); document.write("Küçük olan=" + c);</pre>	Küçük olan=10

Yukarıdaki örnekte, 10 ve 20 değerleri ile çağrılan fonksiyon, *return x* komutuna gelince x'in değerini (10) *geri* döndürür ve fonksiyon *bu satırda durur*.

Fonksiyonların Çalışma Zamanı

Fonksiyonlar 3 farklı zamanda çalışabilir:

- Komutlar içerisinde *çağırıldıklarında*,

- HTML sayfasındaki bir elemanın herhangi bir *olayı* tetiklendiğinde,
- Kendi kendine.

Bu üç durumdan ilki yukarıdaki örneklerde gösterilmiştir. Fonksiyon tanımlandıktan sonra adı ve argümanlar ile fonksiyon çağrılabılır.

İkinci durumda fonksiyon çağrırmak için HTML etiketlerinin içerişine, o etikete özgü olaylardan herhangi birinin adı ve tanımlanmış olan bir fonksiyonun adı ve argümanlarını uygun biçimde yazmak gereklidir. *Olaylar (events)* konusu bu ünitemin devamında ele alınmıştır. Ancak örnek olması açısından aşağıdaki kodlara bakabilirsiniz.

Tablo 11.26. Tıklanan Paragrafin Metninin Değişmesi



onClick olayı fare ile tıklanma durumunda çalışacak fonksiyonlar için kullanılır.

Javascript Komutları

```
<script>
function yaziDegistir(yeniMetin){
    document.getElementById("p1").innerText=yeniMetin;
}
</script>

<p id="p1" onClick="yaziDegistir('Merhaba');">Bu bir paragraftır</p>
```

İLK ÇIKTI:	TIKLAMA SONRASI ÇIKTI
Bu bir paragraftır	Merhaba

Yukarıdaki örnekte, *paragraf* etiketine *tıklanma* olayında (*onClick*) çağrılmaması için *yaziDegistir()* fonksiyonu yazılmıştır. Bu durumda kullanıcı fare ile bu paragrafin üzerine her tıkladığında *yaziDegistir()* fonksiyonu çalışacaktır.

Üçüncü durum olan *kendi kendine* çalışmada yönteminde ise fonksiyona bir isim verilmeden aşağıdaki gibi yazılır. Burada amaç aslında yeni bir fonksiyon yaratmak değil, bazı komutları ve değişkenleri sanki bir fonksiyonun içeresindeymiş gibi kullanıp, lokal oldukları için fonksiyon sonunda hafızadan silinmelerini sağlamaktır.

Tablo 11.27. Kendi Kendine Çalışan Fonksiyon

Javascript Komutları

```
<p id="bilgi1"></p>
<p id="bilgi2"></p>
<script>
(function(){
document.getElementById("bilgi1").innerText="İlk fonksiyon çalıştı";
})();
function yap(){
document.getElementById("bilgi2").innerText="İkinci fonksiyon çalıştı";}
```

```

    }
    </script>
    <button onclick="yap()">Dene</button>

```

ÇIKTILAR

İlk fonksiyon çalıştı <input type="button" value="Dene"/>	Sayfa yüklenince ilk çıktı
İlk fonksiyon çalıştı İkinci fonksiyon çalıştı <input type="button" value="Dene"/>	“Dene” butonu tıklandıktan sonra çıktı

Yukarıdaki örnekte ilk fonksiyon yazıldığı yerde otomatik olarak çalışır. Sanki bir komutmuş gibi ilk fonksiyonun bloku kapandıktan sonra () sembollerini konur.

Fonksiyonların JS Dosyalarına Yazılması

Fonksiyonlar genellikle birden çok HTML sayfasında benzer biçimde çalışılmaları için yazırlılar. Bu durumda fonksiyonun her HTML sayfasında tekrar tanımlanması gereklidir. Bunun yerine fonksiyonlar *JS uzantılı Javascript dosyaları* içeresine yazılıp, HMTL sayfalarına *eklenebilirler*.

Tablo 11.28. Fonksiyonların Javascript Dosyalarında Saklanması



Verimliliği ve güvenliği artırmak için Javascript fonksiyonları harici JS dosyaları içerisinde tanımlanır.

Javascript Komutları

```

<!-- index.html dosyası -->
<html>
<head>
<meta charset="utf-8">
<title>Untitled Document</title>
<script language="JavaScript" src="scripts.js"></script>
</head>
<body>
<p id="p1" onclick="yaziDegistir('Merhaba', 'p1');">Bu bir paragraftır</p>
</body>
</html>

```

ÇIKTI

```

// scripts.js dosyası

function yaziDegistir(yeniMetin, paragrafID){
    document.getElementById(paragrafID).innerText=yeniMetin;
}

```

İLK ÇIKTI:

Bu bir paragraf



TIKLAMA SONRASI ÇIKTI

Merhaba

Yukarıdaki örnekte, *index.html* dosyasına, aynı klasörde bulunan *scripts.js* dosyasının eklendiği satırı görebilirsiniz. Scripts.js dosyası içerisindeki fonksiyon bu sayede farklı HTML dosyalarında çalışabilir.

Yukarıdaki son örnekte dikkat edilmesi gereken bir başka durum ise fonksiyonun iki parametreli olmasıdır. İkinci parametre, *ID'si* verilen her türlü HTML etiketi ile çalışabilir.

OLAYLAR



Olaylar, Javascript'in HTML ile iletişiminde çok önemlidir.

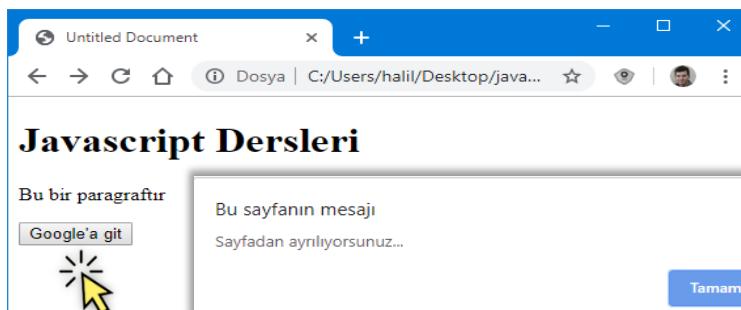
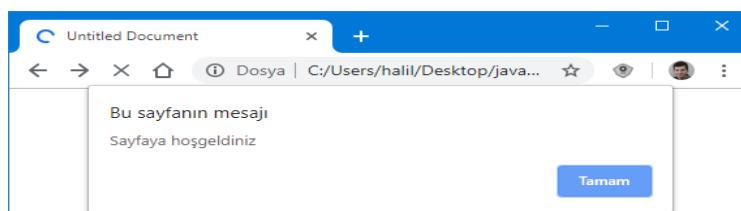
HTML etiketleri sayfaya yüklenme anında ya da yüklenikten sonra kullanıcının hareketlerini algılayabilirler. Algılayabildikleri bu durumlara *olay* (event) denir. Algılanan bir *olay* sonucunda Javascript komutları ya da fonksiyonları çalıştırılabilir. Aşağıdaki kodlara ve devamındaki çıktı ekranlarına bakınız.

Tablo 11.29. Farklı Etiketlerin Farklı Olayları

Javascript Komutları

```
<script>
function git(adres){
    alert("Sayfadan ayrıliyorsunuz...");
    document.location.href=adres;
}
</script>
</head>
<body onload="alert('Sayfaya hoşgeldiniz');">
<h1 onclick="alert('başlığa tıkladınız');">Javascript Dersleri</h1>
<p onclick="alert('başlığa tıkladınız');">Bu bir paragraftır</p>
<button onclick="git('http://www.google.com.tr')">Google'a git</button>
</body>
```

Çıktı aşağıdaki gibi olacaktır.



Şekil 11. 2. Sayfanın Yüklenme Ve Butonun Tıklanması Sonrasındaki Görüntüsü

Yukarıdaki kodlara ve ekran görüntülerine bakıldığında, bazı olayların (**onload**) etiketin yüklenmesi sırasında otomatik olarak, bazı olayların ise **fare ile tıklanma** (**onclick**) olayı gerçekleştiğinde algılama yapıldığı anlaşılabılır. Algılanan olayın sonuna **eşittir** ifadesi ile Javascript komutları ya da fonksiyonları yazılabilir.

Olaylar **HTML** dilinin bir parçasıdır, bu nedenle yazım kuralları Javascript'ten farklı biçimde **büyük küçük harf ayımı** olmadan yazılabilir. Ancak algılanan olayın sonucunda çalıştırılacak Javascript komut ya da fonksiyonları küçük ve büyük harfe duyarlıdır.

```
<button onclick="git('http://www.google.com.tr')">Google'a git</button>
<button ONCLICK="git('http://www.google.com.tr')">Google'a git</button>
```

Şekil 11. 3. Olayların Farklı Biçimde Yazımı Geçerlidir.

En Sık Kullanılan Olaylar

HTML etiketlerinin hepsinin kendisine has algılayabildikleri olay listesi vardır. Bu çok uzun bir listedir. Bunun yerine sıklıkla kullanılan aşağıdaki olayları bilmek çoğu durumda geliştiricilere yeterli olacaktır.

Tablo 11.30. En Çok Kullanılan Olaylar

Olay Adı	Açıklaması
onClick	HTML elemanına kullanıcının fare ile tıklanması
onDblClick	Elemana fare ile çift tıklanması
onLoad	Body, image veya frame gibi elemanların yüklenmesinin tamamlanması
onUnLoad	Web sayfasından çıktığında (tarayıcıyı kapatarak ya da yeni bir sayfaya giderek)
onFocus	Bir elemana odak (focus) yapıldığında
onBlur	Odağa sahip elemanın odağı kaybetmesi sırasında
onMouseOver	Fare işaretçisinin elemanın üzerine geldiğinde
onMouseOut	Fare işaretçisinin elemanın üzerinden çekilmesi sırasında
onMouseMove	Fare işaretçisiyle elemanın üzerinde hareket edilmesinde (elemanın dışına çıkmadan)
onMouseDown	Fare işaretçisinin elemanın üzerindeyken farenin sol tuşunun basılması
onMouseUp	Fare işaretçisinin elemanın üzerindeyken farenin sol tuşunun bırakılması
onKeyDown	Klavyeden bir tuşa basılması
onSelect	Bir metin kutusu (textarea ya da input) içindeki metnin fare ile seçilmesi
onChange	Bir listeden seçim yapıldığında
onResize	Tarayıcı penceresinin boyutunun değiştirilmesi



Olaylar HTML içerisinde büyük/küçük harfle yazılabilir.



Farklı HTML elemanları farklı olayları destekler.

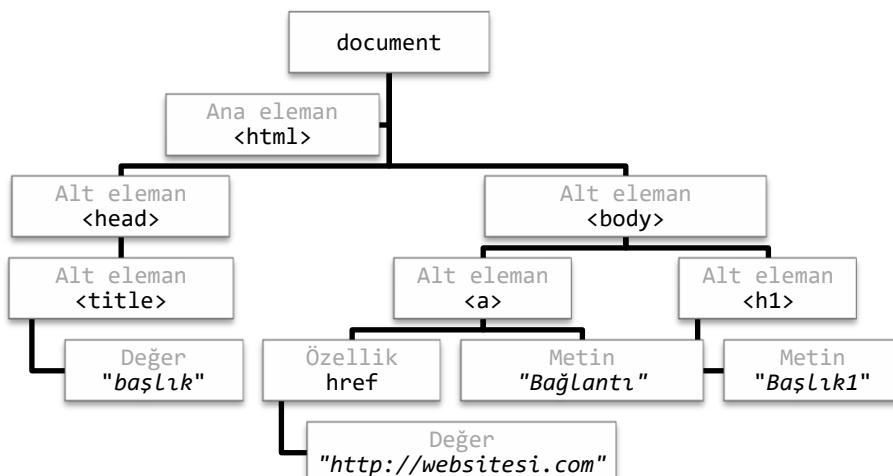


Bireysel Etkinlik

- Yukarıdaki olaylara ilaveten farklı HTML elemanlarına ait onlarca olay vardır. Bu kitap kapsamında olmayan bu olaylara şu adresten ulaşabilirsiniz:
https://www.w3schools.com/jsref/dom_obj_event.asp

DOM MODELİ

HTML sayfaları yüklendiğinde, tarayıcılar sayfadaki tüm elemanları (HTML etiketlerini) **document** ismindeki bir nesnenin alt nesneleri olarak belirler. Bu hiyerarşi sayesinde HTML sayfasındaki tüm etiketlerin içeriği, özellikleri, CSS stilleri ve alt etiketleri değiştirilebilir. Bu hiyerarşik yapıya **Document Object Model** (DOM) denir.



Şekil 11.4. Örnek Dom Hiyerarşisi

DOM **tarayıcı** tarafından oluşturulur ve farklı betik dilleri ile (örneğin Javascript, VBScript vb.) programlanabilir. Bu kitapta standart hâline gelmiş olan Javascript ile DOM programlama gösterilmektedir.

Yukarıdaki şekilde gösterilen hiyerarşik yapı, aşağıdaki HTML kodlarından oluşan bir sayfaya ait olabilir.

Tablo 11.31. Basit Bir HTML Sayfası Kodları Ve Görüntüsü



DOM, farklı betik (script) dilleri ile kullanılabilir, ancak HTML5 standartlarında Javascript kabul edilmiştir.

Javascript Komutu	Çıktı
<pre><html> <head> <title>başlık</title> </head> <body> Bağlantı <h1>Başlık1</h1> </body></pre>	

</html>

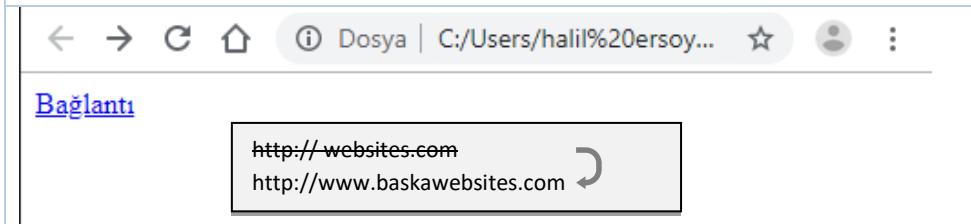
Javascript ile DOM kullanılarak aşağıdaki işlemler yapılabilir:

- Herhangi bir HTML elemanı sayfaya eklenebilir, var olanlar silinebilir.
- Herhangi bir HTML etiketinin herhangi bir özelliği değiştirilebilir, eklenebilir, silinebilir.
- Herhangi bir HTML etiketinin CSS stilleri değiştirilebilir.
- Herhangi bir HTML elemanın olayına yönelik kod yazılabilir.
- Herhangi bir HTML etiketine yeni olay eklenebilir.

DOM, Javascript veya farklı betik dilleri ile dinamik içerik oluşturulabilmesi için;

- HTML etiketlerini *nesne* olarak,
- Bu nesnelerin *özelliklerini*,
- Nesne ve özelliklere ulaşmak için gerekli *metotları* ve
- Nesnelerin *olaylarını* tanımlayan modeldir.

Tablo 11.32. Javascript İle Bir Bağlantının Adresinin Değiştirilmesi

Javascript Komutları	
<body> Bağlantı <script> document.getElementById("link1").href="http://www.baskawebsites.com"; </script> </body>	
ÇIKTI	
 A screenshot of a web browser window. The address bar shows 'Dosya C:/Users/halil%20ersoy...'. Below it, a link labeled 'Bağlantı' is shown. A tooltip or callout box points to this link, displaying two URLs: 'http://websites.com' and 'http://www.baskawebsites.com'. The browser interface includes standard navigation buttons (back, forward, home) and other toolbar icons.	

Yukarıdaki *<a>* etiketinin adresi Javascript ile değiştirilmiştir. Kodlarda yer alan;

- *document*, tüm sayfayı temsil eden *nesne*;
- *getElementById("link1")*, id özelliği "link1" olan alt eleman nesnesine ulaşmak için kullanılan *metot*;
- *href*, erişilen nesnenin bir *özellik*;
- "http://www.baskawebsites.com" ise bu özelliğe verilen yeni *değerdir*.

DOM ile Elemanlara Erişim

DOM'a göre oluşturulan elemanlara erişmek için 4 farklı metot vardır. Bunlar aşağıdaki tabloda gösterilmiştir.



Javascript komutu olan *getElementById* komutundaki büyük ve küçük harflere dikkat edilmeli ve aynen yukarıdaki gibi yazılmalıdır.

Tablo 11.33. Elemanlara Erişim Yolları

Javascript Komutu	Açıklama
document.getElementById("idBilgisi")	Sayfa içerisinde <i>id</i> özelliğine göre elemana erişme
document.getElementsByName("nameBilgisi")	Sayfa içerisinde <i>name</i> özelliğine göre elemanlara erişme
document.getElementsByTagName("EtiketAdı")	Sayfa içerisinde <i>etiketin adına</i> göre elemanlara erişme
document.getElementsByClassName("classAdı")	Sayfa içerisinde <i>CSS sınıfı</i> özelliğine göre elemanlara erişme

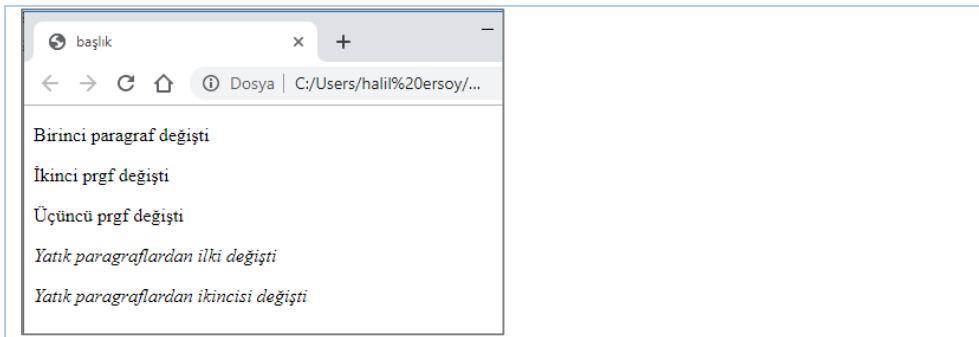
Yukarıdaki dört erişim yolundan *getElementById()*, sayfadaki *id* bilgisi bilinen *bir elemana* erişir. Diğer metotlar ise, *name özelliği*, *sınıf özelliği* veya *etiket adı* verilen *birden çok elemana* erişmeyi sağlar. Bu son üç metot birden fazla eleman bulunması ihtimaline karşı, elemanları 0'dan başlayan *dizinin elemanları* olarak erişir. Aşağıdaki örnekte, *getElementsByName()* ve *getElementsByClassName()* metotlarından sonra dizinin 0'dan başlayan indis numarası bu nedenle kullanılmıştır.

Tablo 11.34. Elemanlara Erişim Ve Çıktı

Javascript Komutları
<pre><head> <style> .yaziStili{ font-style: italic; } </style> </head> <body> <p id="yazi1">Bu birinci paragraftır.</p> <p name="yazi2">Bu ikinci paragraftır.</p> <p name="yazi2">Bu üçüncü paragraftır.</p> <p class="yaziStili">Bu dördüncü paragraftır.</p> <p class="yaziStili">Bu beşinci paragraftır.</p> <script> document.getElementById("yazi1").innerText="Birinci paragraf değişti"; document.getElementsByName("yazi2")[0].innerText="İkinci prgf değişti"; document.getElementsByName("yazi2")[0].innerText="Üçüncü prgf değişti"; document.getElementsByClassName("yaziStili")[0].innerText="Yatık paragraflardan ilki değişti"; document.getElementsByClassName("yaziStili")[1].innerText="Yatık paragraflardan ikincisi değişti"; </script> </body></pre>
ÇIKTI



Elemanlara erişim için kullanılan metotlar, eğer kritere uyan eleman bulunamazsa *undefined* sonucunu geri döndürür.



InnerHTML ve InnerText Özellikleri

Çoğu HTML etiketi kendi içlerinden başka etiketleri ya da metni barındırır. Eğer bir HTML elemanın açılış ve kapanış etiketlerinin içine **yenİ HTML etiketleri** eklemek isterseniz ya da var olan HTML etiketlerini değiştirmek isterseniz, **innerHTML** özelliğini kullanınız. Öte taraftan eğer bir HTML elemanın açılış ve kapanış etiketleri arasına **metin** eklemek ya da var olan metni değiştirmek isterseniz, **innerText** özelliğini kullanınız.



Bir eleman içeresine
yenİ eleman eklenmek
istenirse **innerHTML**
özellikİ kullanılmalıdır.

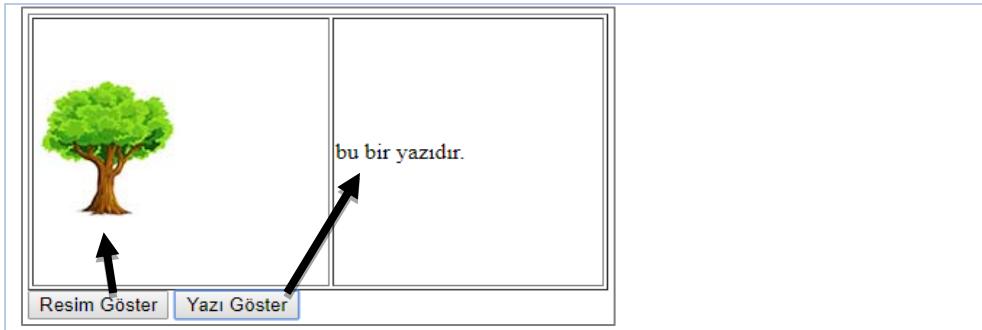
Tablo 11.35. Innerhtml Ve Innertext Özelliği

Javascript Komutları

```
<script>
function resimYukle(resimDosyasiAdi){
    var resimHTMLKodlari=<img src="" + resimDosyasiAdi + "\"/>";
    document.getElementById("hucre1").innerHTML=resimHTMLKodlari;
}
function yazıYukle(metin){
    document.getElementById("hucre2").innerText=metin;
}

</script>
</head>
<body>
<table border="1" width="400" height="200">
<tr>
    <td id="hucre1"></td>
    <td id="hucre2"></td>
</tr>
</table>
<button onclick="resimYukle('resim1.jpg');">Resim Göster</button>
<button onclick="yazıYukle('bu bir yazıdır.');">Yazı Göster</button>
</body>
```

ÇIKTI



DOM Elemanlarının Özellik ve Stillerini Değiştirmek



Bir elemanın stil özelliğini değiştirmek için **setAttribute** metodu ya da o özelliğin adının açıkça yazılmış şekli kullanılabilir.

HTML etiketlerinin özellik (attribute) ve stil öğelerini (style) Javascript ile aşağıdaki gibi değiştirebilirsiniz. Değiştirebileceğiniz tüm CSS kurallarının listesine https://www.w3schools.com/jsref/dom_obj_style.asp adresinden ulaşabilirsiniz.

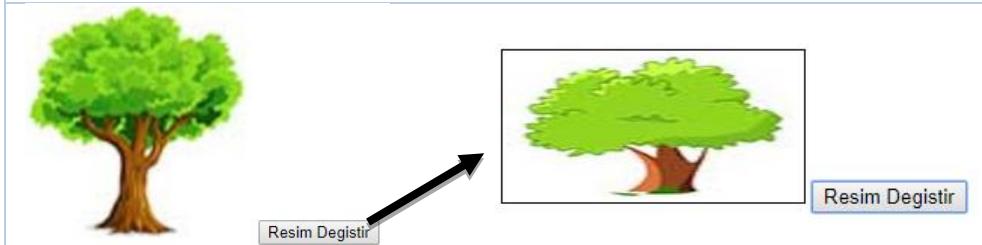
Tablo 11.36. HTML Etiketlerinin Özellik Ve Stillerinin Değiştirilmesi

Javascript Komutları

```
<script>
function resimDegistir(resimDosyasiAdi){
document.getElementById("resimKutusu").src=resimDosyasiAdi;
document.getElementById("resimKutusu").height=100;
// Üstteki satır VE YA alttaki satır
document.getElementById("resimKutusu").setAttribute("height",100);
document.getElementById("resimKutusu").style.border="1px solid black";
}
</script>
</head>
<body>

<button onclick="resimDegistir('resim2.jpg');">Resim Degistir</button>
</body>
```

ÇIKTI



Yukarıdaki gibi herhangi bir stil öğesini değiştirmenin ötesinde, elemanın varsa CSS sınıfını **className** özelliğine yeni sınıf atayarak aşağıdaki gibi değiştirebilirsiniz:



Javascript ile değiştirilecek olan stil sınıfları önceden tanımlanmış olmalıdır.



Örnek

```
•document.getElementById("resim").className="kucukResimSınıfı";
```

WINDOW NESNESİ

Window nesnesi, kullanıcının **tarayıcısının** özelliklerinin toplandığı ve Javascript ile programlanabilir alt özellik ve metotlara sahip nesnedir. Bu bölümde Javascript ile sıkılıkla yapılan bazı tarayıcı işlemlerini, **window** nesnesinin metod ve özelliklerini kodlayarak ele alacağız.

Tarayıcı Ebatları

Kullanıcıların **tarayıcıları** farklı **ekranlarda** ve farklı en ve boy değerlerinde olabilir. Bu durum Javascript ile algılanabilir.

Tablo 11.37. Tarayıcının Ve Ekranın Ebatlarını Öğrenme

Javascript Komutu	Açıklama
window.innerWidth	Tarayıcının genişliğini piksel cinsinden verir.
window.innerHeight	Tarayıcının yüksekliğini piksel cinsinden verir.
window.screen.width	Ekranın genişliğini piksel cinsinden verir.
window.screen.height	Ekranın yüksekliğini piksel cinsinden verir.
window.screen.pixelDepth	Ekranın renk derinliğini (16,24 veya 32) verir.

Yukarıdaki özellikler farklı ekran büyütüklerinde farklı görünümlü HTML sayfalarını göstermek için kullanılabilir.

Tablo 11.38. Tarayıcı Ve Ekran Nesnelerinin Özellikleri



Tarayıcının ebatları kullanıcı tarafından değiştirilirse, **body** elemanın **onresize** olayı tetiklenmiş olur.

Javascript Komutları

```
<p id="bilgi"></p>
<script>
var eleman=document.getElementById("bilgi");
bilgi.innerHTML="Tarayıcı Genişliği = " + window.innerWidth +
"px<br>";
bilgi.innerHTML=bilgi.innerHTML+"Tarayıcı Yüksekliği=" +
window.innerHeight+"px";
bilgi.innerHTML=bilgi.innerHTML+"<hr>";
bilgi.innerHTML=bilgi.innerHTML+"Ekran Genişliği=" +
window.screen.width + "px<br>";
bilgi.innerHTML=bilgi.innerHTML+"Ekran Yüksekliği=" +
window.screen.height+"px";
bilgi.innerHTML=bilgi.innerHTML+"<hr>";
bilgi.innerHTML=bilgi.innerHTML+"Ekran Çözünürlüğü= " +
window.screen.pixelDepth+ "bit<br>";
bilgi.innerHTML=bilgi.innerHTML+"<hr>";
</script>
```

ÇIKTI

Tarayıcı Genişliği = 500px
Tarayıcı Yüksekliği = 189px

Ekrان Genişliği = 1920px
Ekrان Yüksekliği = 1080px

Ekrان Çözünürlüğü= 24bit

Tarayıcı Adresi

Window.location nesnesi, ziyaret edilen **web sitesinin** ve **sayfasının** tam adresini, sayfanın adını, bağlantı protokolünü verir. İstenirse başka bir web sayfası adresi verilerek yeni sayfanın **yüklenmesi** sağlanır.

Tablo 11.39. Window.Location Nesnesinin Özellikleri Ve Çıktıları

Javascript Komutu	Açıklama
window.location.href	Aktif sayfanın tam adresini verir. Örnek çıktı: http://www.websites.com.tr/javascript/konu1.html
window.location.hostname	Aktif sayfanın sunucu ismini verir. Örnek çıktı: http://www.websites.com.tr
window.location.pathname	Aktif sayfanın yolunu ve dosya adını verir. Örnek çıktı: /javascript/konu1.html
window.location.protocol	Aktif sayfanın bağlantı protokolünü verir (http: ya da https: şeklinde). Örnek çıktı: http://www.websites.com.tr
window.location.assign()	Yeni sayfanın yüklenmesini sağlar. Örnek kullanımı: window.location.assign("http://www.site2.com");

Tarayıcı Geçmiş Nesnesi

Tarayıcılar ziyaret edilen tüm sayfaları “geçmiş” adı altında bir liste şeklinde saklar. Bu listeye Javascript ile ulaşabilirsiniz. Örneğin kullanıcınızın herhangi bir işlemi yanlış yaptığında bir önceki sayfaya gitmesini sağlayabilirsiniz. Bunun için **window.history** nesnesinin **back()** ve **forward()** metotları kullanılır.

Tablo 11.40. Tarayıcı Geçmiş Nesnesi

Javascript Komutları
<button onclick="window.history.back();">Geri git</button>
<button onclick="window.history.forward();">İleri git</button>
ÇIKTI
Geri git İleri git

Tarayıcı Adı ve Sürümünü Belirleme

Javascript'in en çok kullanıldığı yer, tarayıcı algılamaktır. Bunun nedeni, farklı tarayıcıların farklı kodları (Javascript, CSS ve hatta HTML) farklı biçimde desteklemesidir. Geliştiriciler bu probleme karşın, farklı tarayıcılar için alternatif kodlar yazarak yapılması gereken işlemlerin farklı tarayıcıya sahip kullanıcıları için yapılmasını sağlamaya çalışırlar. Bunun için öncelikle tarayıcının adının ve sürüm numarasının algılanması gereklidir.

Window.navigator nesnesi, tarayıcı hakkında bilgi veren bir nesnedir. Aşağıdaki örnekte tarayıcı adı ve sürümünü algılayan kodlar yer almaktadır.

Tablo 11.41. Tarayıcı Algılama

Javascript Komutları

```
<p id="bilgi"></p>
<script>
var eleman=document.getElementById("bilgi");
bilgi.innerHTML="navigator.appName = "+window.navigator.appName + "<hr>";
bilgi.innerHTML+="navigator.appVersion= " +
    window.navigator.appVersion+"<hr>";
bilgi.innerHTML+="navigator.platform= "+window.navigator.platform+"<hr>";
var strVersion=window.navigator.appVersion;
var sonuc="";
if(strVersion.indexOf("OPR") > -1) sonuc="Opera";
else if(strVersion.indexOf("Edge") > -1) sonuc="Edge";
else if(strVersion.indexOf("Chrome") > -1) sonuc="Chrome";
else if(strVersion.indexOf("rv:11") > -1) sonuc="Internet Explorer";
bilgi.innerHTML += "Sonuç = " + sonuc + "<hr>";
</script>
```

ÇIKTI

```
navigator.appName = Netscape
navigator.appVersion = 5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/76.0.3809.100
Safari/537.36
navigator.platform= Win32
Sonuç = Chrome
```



Google Chrome
Çıktısı



Bireysel Etkinlik

- Yukarıdaki kodu yazıp, farklı tarayıcılarda deneyiniz.
- Sadece belirli bir tarayıcıda çalışacak kodlar ekleyiniz.

Yukarıdaki örneğe baktığımızda `navigator` nesnesinin `.appName`, `.appVersion` ve `.platform` özelliklerinin çıktılarının farklı tarayıcılarda da olsa benzer ya da aynı çıktıları verdiğiğini görüyoruz. Bu durum, tarayıcının tam adını ve sürümünü algılamak için ek adımlar atmamız gerektir. Örneğin tarayıcının versiyon bilgisinde farklı ifadeleri `arayarak` yukarıdaki örnekteki gibi kesin sonuç elde edebiliriz.

Mesaj Kutuları

Kullanıcıya tarayıcının içerisinde açılarak kısa mesajlar vermeyi veya kullanıcıdan kısa bilgiler almayı sağlayan birkaç mesaj kutusu alternatifleri vardır. Bunlar `window` nesnesinin;

- `window.alert()` komutu
- `window.confirm()` komutu ve
- `window.prompt()` komutudur.

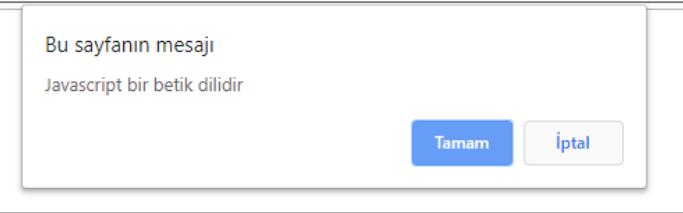
Tablo 11.42. Alert Metodu Örneği

Javascript Komutları	
<script>	alert("Javascript ünitesine hoş geldiniz.");
ÇIKTI	
	


Alert(), confirm() ve prompt() metotları, **window** kelimesi olmadan da kullanılabilir.

Alert() metodu ile ister örnekteki gibi açık ifadeler, istenirse de farklı değişkenlerin değerleri ekrana yazdırılabilir. Mesaj kutusunda görülen “Tamam” butonunun üzerindeki yazı, renk ve büyütülebilir. Farklı tarayıcılarda farklı görülebilir.

Tablo 11.43. Confirm Metodu Örneği

Javascript Komutları	
<p id="bilgi"></p>	<script>
	if(confirm("Javascript bir betik dilidir"))
	metin="Tamama basıldı";
	else
	metin="İptale basıldı"; document.getElementById("bilgi").innerText=metin;
	</script>
ÇIKTI	
	

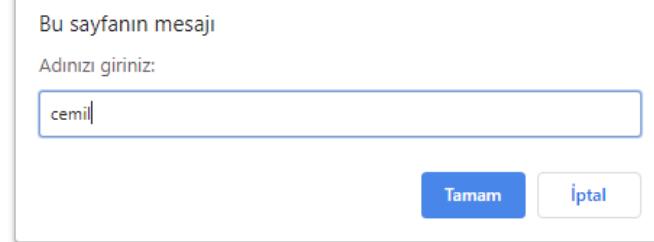
Yukarıdaki örnekte görülen **confirm()** metodunda, ekrana yazılan mesajdan başka ayrıca “**Tamam**” ve “**İptal**” butonları da vardır. Kullanıcının bu iki butondan herhangi birine tıklamasıyla pencere kapanır. Eğer “Tamam” butonuna tıklanırsa **confirm** metodу **true**, “İptal” butonuna tıklanırsa “**false**” değerini döndürür. Örnekteki gibi bir **if** ile bu değer kontrol ettirilip, farklı işlemler kullanıcı cevabına göre yapılabilir.

Tablo 11.44. Prompt Metodu Örneği

Javascript Komutları	
<p id="bilgi"></p>	<script>
	var isim=prompt("Adınızı giriniz:", "adınız");
	if(isim!= null)
	metin="Merhaba sayın " + isim;

```
else  
    metin="İptale basıldı";  
document.getElementById("bilgi").innerText=metin;  
</script>
```

ÇIKTI



Prompt komutunda ilk parametre olarak ekranada görünmesini istediğiniz ifadeyi, ardından ikinci parametre olarak da metin kutusunda varsayılan değer olarak yer almasını istediğimiz değeri yazılır. Kullanıcı bilgi girip "Tamam" butonuna tıklarsa, bu metot *girilen değeri*; eğer "İptal" butonuna tıklarsa *null* değerini döndürür.

Açıılır Pencere Açma

Birçok web sitesinde var olan tarayıcının içerisinde yeni bir sayfa ya da sekme açmak için *window.open()* metodu kullanılır. Bu metot toplam dört parametre alabilir.



Açıılır pencereler (*popup windows*) bazı tarayıcılar tarafından engellenebilir.

```
window.open("http://yenisite.com", "_blank", "width=200", true);
```

Şekil 11.5. Window.Open Metodu

Yukarıdaki örnek gösterimde, ilk parametre açılmasını istediğimiz pencerenin içerisinde görüntülenecek olan web sayfasının adresidir.

İkinci parametre ise açılacak olan pencerenin, var olan pencere olmayıp yeni bir pencere olacağıdır. Bu parametrenin alternatifleri şunlardır:

- *_blank* → yeni tarayıcı penceresi veya yeni tarayıcı sekmesi
- *_parent* → varsa ana frame içerişi
- *_self* → şu anda içinde bulunan pencere ya da frame
- *_top* → Varsa tüm frameset'lerin üstündeki ana pencere
- *pencereAdı* → Varsa adı pencereAdı olan frame içerişi

Üçüncü parametre, açılacak pencerenin genişlik, yükseklik, ekrandaki konumu gibi özellikleri parametrik olarak belirlemek için kullanılır. Bununla ilgili detaylı ayarlara https://www.w3schools.com/jsref/met_win_open.asp sayfasından ulaşabilirsiniz.



true ve *false* ifadeleri tek başına bir değerdir, metin türü değildir!

Dördüncü parametre ise true ya da false olabilir. *True* olduğunda açılan pencerenin tarayıcı geçmişine eklendikten sonra var olan sayfanın geçmişten çıkarılmasını sağlar. *False* değeri ise tarayıcı geçmişine hem kendisini ekler hem de var olan sayfanın kalmasını sağlar.



:Özet

• Bu üitede Javascript dili ile yapılabilecek ileri düzey işlemler için gerekli konular ele alınmıştır. Bunlar metin (string) işlemler, döngüler, fonksiyonlar, olaylar, DOM hiyerarşisi ve Window nesnesinin özellikleridir.

• METİN İŞLEMLERİ

• Metin tür değerler (string) karakter dizileri şeklinde hafızada tutulur. Metinlerin her bir karakterine o karakterin indis numarası ile ulaşılabilir. İndis numarası 0'dan başlar. Metnin karakter sayısı ise length özelliği ile elde dilir. Bunun dışında metin üzerinde yapılabilecek işlemler hazır fonksiyonlar ile yapılır.

• DÖNGÜLER

• Döngüler aynı işlemin birden fazla yapılması gerekiğinde komutların blok halinde yazılıp döngü komutu ile kaç kez tekrarlanmasıın belirlendiği yapılardır. For, while ve do-while olmak üzere 3 farklı döngü komutu ile işlemler yapılabilir. Dizilerin elemanlarına ulaşmak için genellikler for döngüsü kullanılır.

• FONKSİYONLAR

• Fonksiyonlar, birbiri ile ilgili olduğu düşünülen komutların gruplanarak tek bir komut benzeri yapılar içerisinde saklanması sağlanan yapılardır. Fonksiyonlar öncelikle tanımlanır. Tanımlamada fonksiyon ismi ve istenirse parametreleri yazılır. Küme parantezleri içerisinde ise çalıştırılmak istenen komutlar yazılır. Fonksiyonlar bu yolla tanımlandıkları yerde ve zamanda değil, daha sonra çağrıldıkları zaman çalışır. Çağırıldığında fonksiyon adı ve varsa parameterleri (argümanı) yazılır.

• Eğer fonksiyon işlem sonucunda bir değer üretip bunu çağrııldığı noktaya geri döndürmek üzere tanımlanırsa, fonksiyon bloğu içinde bu değer return kelimesi ile yazılmalıdır.

• Fonksiyonlar ya çağrıldıkları zaman, ya bir olay gerçekleştiğinde, ya da özel biçimde tanımlanarak tanımlandıkları anda çalışabilirler.

• Parametreler fonksiyon içinde dğışırse bu çağrı çağırıldıkları yere yansımaz. Dizi veya nesne türündeki parametreler ise buna istisna olarak fonksiyon içinde deşirlerse çağrıldıkları yerde de deşir.

• OLAYLAR

• Tüm HTML etiketlerinin kullanıcının bir takım hareketlerini veya sayfanın yüklenme zamanı algılayıp cevap verebileceği olayları (events) vardır. Bu olaylar farklı etiketler için farklı sayıda ve isimdedir. Bu oylara eğer bir Javascript komutu ya da fonksiyonu yazılsa, olay gerçekleştiğinde bu komut ya da fonksiyon çalışır.

• En çok kullanılan olaylar onclick, onmouseover, onload olaylarıdır.

• DOM MODELİ

• HTML sayfaları yükleniğinde, tarayıcılar sayfadaki tüm elementleri (HTML etiketlerini) *document* ismindeki bir nesnenin alt nesneleri olarak belirler. Bu hiyerarşide HTML sayfasındaki tüm etiketlerin içeriği, özellikleri, CSS stilleri ve alt etiketleri değiştirilebilir. Bu hiyerarşik yapıya *Document Object Model (DOM)* denir.



Özet(devam)

•DOM ile Elemanlara Erişim

- Tarayıcı tarafından yaratılan ilk nesne **document** nesnesidir. HTML sayfasındaki diğer etiketler (elemanlar) bu nesnenin alt elemanları olarak hafızada tutulur. Bu hiyerarşide herhangi bir etikete erişmek istenirse öncelikle o etiketin "id" özelliğine bir isim verilmeli, sonrasında **document.getElementById()** metodu ile o elemana erişim sağlanabilir.
- **GetElementById()** metodu ile erişilen elemanın kendine has özellik ve metodları vardır. Bu metod ve özellikler Javascript ile değiştirilebilir.

•InnerHTML ve InnerText Özellikleri

Bir HTML elemanın açılış ve kapanış etiketlerinin *arasına yeni HTML etiketleri* eklemek isterseniz ya da var olan HTML etiketlerini değiştirmek isterseniz, *innerHTML* özelliğini kullanınız. Öte taraftan eğer bir HTML elemanın açılış ve kapanış etiketleri *arasına metin* eklemek ya da var olan metni değiştirmek isterseniz, *innerText* özelliğini kullanınız.

•DOM Elemanlarının Özellik ve Stillerini Değiştirmek

HTML etiketlerinin özellik (attribute) ve stil öğelerini (style) Javascript ile aşağıdaki gibi değiştirebilirseniz.

```
document.getElementById("resim").style.border="1px solid black";  
document.getElementById("resim").className="kucukResimSınıfı";
```

•WINDOW NESNESİ

- Tarayıcı ve var olan web sayfası hakkında bilgi almak, kullanıcıya tarayıcı içerisinde mesajlar vermek ve kimi zamanda kullanıcılardan kısa bilgiler almak için window nesnesinin alt nesneleri ve metodları kullanılır.
- **Window.open()** metodu ile aılır pencereler açılabilir.

DEĞERLENDİRME SORULARI

1. Aşağıdakilerden hangisi metnin içindeki harfin küçük hâlini geri döndüren fonksiyondur?
 - a) toSmallLetter()
 - b) toSmallString()
 - c) toLowerString()
 - d) toLowerCase()
 - e) toCaseMin()

2. Aşağıdaki iki satır sonucunda, b değişkeninin değeri ne olur?

```
a="merhaba";
b=a.indexOf("a");
```

 - a) true
 - b) 7
 - c) 4
 - d) 2
 - e) 6

3. Aşağıdaki for döngülerinden hangisi yazım hatası yoktur?
 - a) for(s < 20; s++) { a += s; }
 - b) for(s = 1 , s < 20 , s++) { a += s; }
 - c) for(s = 1 ; (s < 20) ; (s++); { a += s; }
 - d) for(1<s<20; s++) { a += s; }
 - e) for(s = 1 ; s < 20; s++) { a += s; }

4. Aşağıdaki while döngülerinden hangisi sonsuz döngüdür?
 - a) t=0; s=1; while(s>10){ t += s; }
 - b) t=0; s=1; while(s<10){ t += s; }
 - c) t=0; s=2; while(s<5){ t += s; s++; }
 - d) t=0; s=20; while(s>10){ t += s; s--; }
 - e) t=0; s=20; while(s>5){ t += s; s=s-2; }

5. Aşağıdaki dizi tanımlamasından sonra, dizinin uzunluğunu ekrana yazan komut hangisidir?

```
var dizi=["TL", "USD", "EU"];
```

 - a) document.write(dizi.items.count);
 - b) document.write(dizi.length);
 - c) document.write(dizi[].values);
 - d) document.write(count(dizi));
 - e) document.write(dizi.index);

6. Aşağıdakilerden hangisi indexOf komutunu doğru biçimde tanımlar?
 - a) Dizi indislerini değiştirir.
 - b) Diziyi sıralar.
 - c) Diziyi ters çevirir.
 - d) Dizi içinde arama yapar.
 - e) Dizi eleman sayısını verir.
7. Kendisine gönderilen 2 sayının farkını bulup geri döndüren fonksiyon hangi seçenekte doğru olarak tanımlanmıştır?
 - a) function f(a,b){return a-b;}
 - b) function f(a,b){f=a-b;}
 - c) function f(a,b){a-b; return;}
 - d) function f(){ a-b;}
 - e) function f(){return a-b;}
8. Bir web sayfası yüklenliğinde çalışan olay aşağıdakilerden hangisidir?
 - a) onLoad
 - b) onClick
 - c) onMouseOver
 - d) onVisit
 - e) onFocus
9. Aşağıdakilerden hangisi CSS sınıfı “uyarı” olan elemanların tümünü getirir?
 - a) document.getElementById(“uyarı”)
 - b) document.getElementsByTagName(“uyarı”)
 - c) document.getElementsByClassName(“uyarı”)
 - d) document.getElementsByName(“uyarı”)
 - e) document.getElementsByStyleName(“uyarı”)
10. Aşağıdakilerden hangisi ziyaret edilen web sayfasının tam adresini verir?
 - a) window.location.adress
 - b) window.location
 - c) window.location.port
 - d) window.location.hostname
 - e) window.location.href

Cevap Anahtarı

1.d, 2.c, 3.e, 4.b, 5.b, 6.d, 7.a, 8.a, 9.c 10 -

YARARLANILAN KAYNAKLAR

Javascript Dersleri. 7 Temmuz 2019 tarihinde <http://javascript.sitesi.web.tr/> adresinden erişildi.

Javascript Tutorial. 23 Temmuz 2019 tarihinde <https://www.w3schools.com/js/> adresinden erişildi.

Javascript HTML DOM Nesneleri. 29 Temmuz 2019 tarihinde <https://www.yazilimbilisim.net/javascript/javascript-html-dom/> adresinden erişildi

JQUERY'E GİRİŞ

İÇİNDEKİLER



- jQuery Nedir?
 - Bir JavaScript Kütüphanesi olarak jQuery
 - Nasıl Çalışır?
- jQuery Kurulumu
- jQuery Sözdizimi (Syntax)
 - Seçiciler (Selectors)
 - Olaylar (Events)
- jQuery Efektleri

HEDEFLER



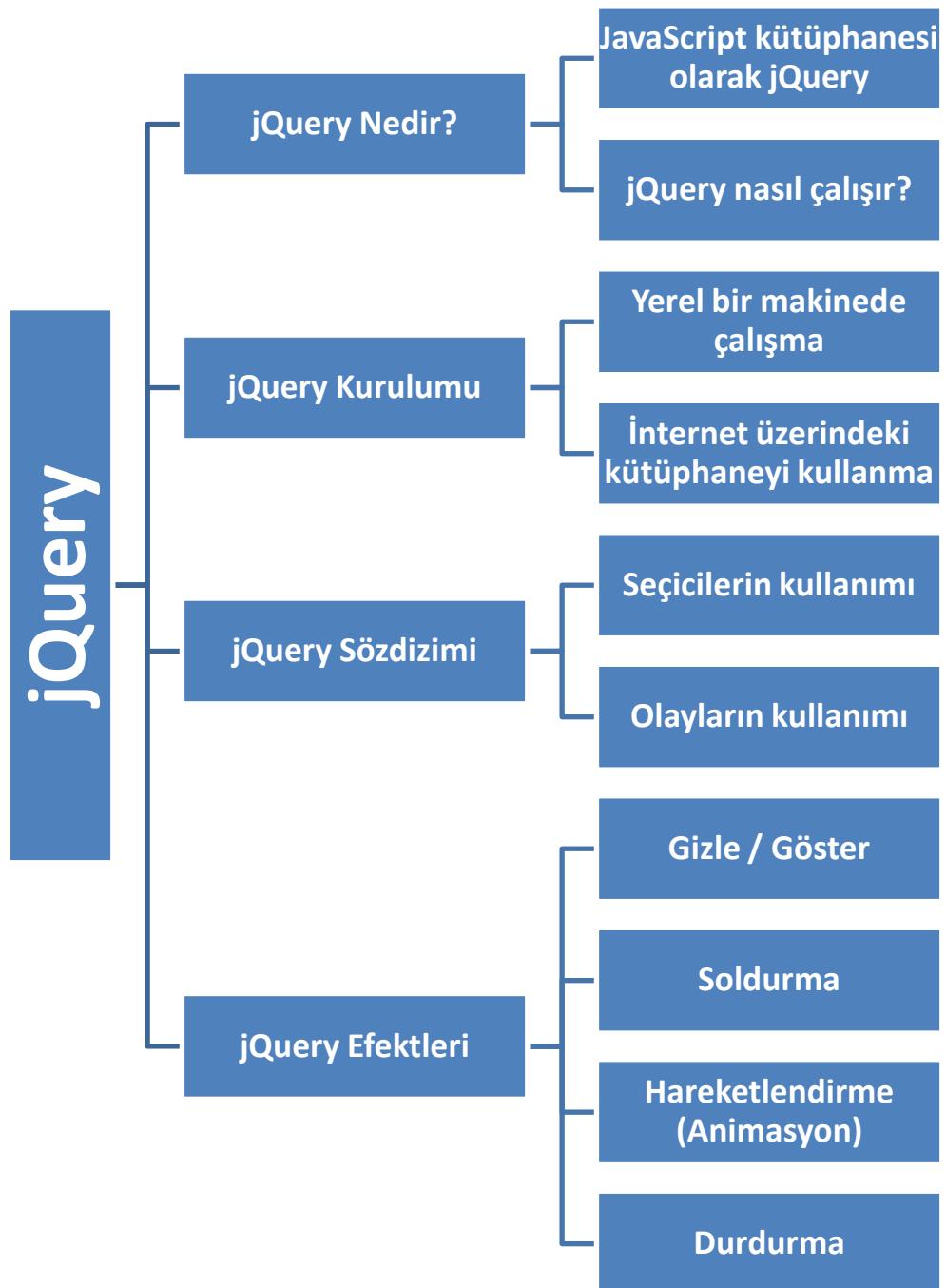
- Bu üniteyi çalışıktan sonra;
- jQuery kütüphanesinin kurulumunu yapabilecek,
- jQuery'nin genel sözdizimi ve kullanımı ile ilgili bilgi sahibi olabilecek,
- jQuery'deki seçicileri kullanabilecek,
- jQuery'de olaylar oluşturabilecek,
- HTML elemanları ile animasyonlar yapabileceksiniz.



Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET
PROGRAMCILIĞI II
Öğr. Gör. Rafet
Orçun MADRAN

ÜNİTE
12



GİRİŞ

Web teknolojileri HTML standartlarının ilk kullanılmaya başlandığı 1993 yılından bu yana inanılmaz bir hızla gelişti. Özellikle farklı hizmet sektörlerinde yer alan birçok uygulamanın İnternet tabanlı olarak kullanılmaya başlaması hem tarayıcıların yeteneklerinin gelişmesine hem de kullanılan teknolojilerde çeşitliliğe yol açtı. Aslında bu gelişmeyi tetikleyen, tarayıcı üzerinde çalışan Internet tabanlı uygulamalardan masaüstü bilgisayarlarda çalışan uygulamaların performansının beklenmesiydi. Son kullanıcılar, bilgisayarlarına ek bir yazılım kurmadan tüm işlemleri İnternet tarayıcıları üzerinden gerçekleştirmek istiyorlardı.

HTML tek başına yukarıda belirttiğimiz uygulama deneyimini son kullanıcıya yaşatabilecek bir altyapıya sahip değildir. Bu noktada devreye giren istemci-sunucu mimarisinin ürettiği çözümler de yeterli gelmemektedir; her bir işlem adımda Web sayfasının yenilenmesine yol açan geleneksel HTML altyapısı, kullanıcıya sunulmak istenen hızı ve esnekliği sağlamamaktadır. Çözüm, Web sayfalarının yenilenmesine gerek kalmadan da veri alışverisini ve etkileşimi mümkün kılacak teknolojilerde yatomaktadır. Ajax teknolojileri olarak ifade edilen bu mimarinin en yaygın kullanılan kütüphanelerinden biri JQuery'dir.

Bu ünite, Web programlama içerisinde ihtiyaç duyulabilecek navigasyon, animasyon ve sayfa içi etkileşimleri oluşturabilmenizi sağlayacak kodları çok kolay bir şekilde yazabilmenizi sağlayacak araçları size sunacaktır. Bu çalışmaları gerçekleştirebilmek için kullanacağımız JQuery kütüphanesi hakkında temel bilgiler, kurulum, kullanım şekilleri ve örnek uygulamalar da yine bu ünite içinde yer olacaktır.

jQuery NEDİR?



jQuery, hızlı, küçük ve zengin özelliklere sahip bir JavaScript kütüphanesidir. Çok sayıda tarayıcıda çalışan kullanımı kolay bir API (Application Programming Interface / Uygulama Programlama Arayüzü) ile HTML belgesinde gezinme, olay işleme, animasyon ve diğer Ajax uygulamalarını çok daha basit hâle getirir. Çok yönlü ve genişletilebilir yapısı çok fazla sayıda geliştirici tarafından tercih edilmesine yol açmaktadır.

jQuery, açık kaynak kodlu bir projedir. Proje ile ilgili tüm bilgilere <http://jquery.com/> adresinden ulaşılabilir. Projenin tüm kodları MIT lisansı ile kullanıma sunulmuştur. MIT lisansının içeriği ve kullanım ile ilgili detaylı bilgi <http://ozgurlisanslar.org.tr/mit/> adresinde yer almaktadır.

Bir JavaScript Kütüphanesi Olarak jQuery

Programlama dillerinin standart yapılarına ek olarak kullanılabilen özelliklerin içinde barındıran kod bloklarını **kütüphane** olarak tanımlayabiliriz. Kütüphaneler belirli bir amaç çerçevesinde birçok tanımlamayı ve hazır fonksiyonu içlerinde bulunduran yapılar olarak kod yazma süreçlerini çok kolaylaştırmakta ve geliştiricilere hızlı uygulama geliştirme olanağı sunmaktadır.

Slogan olarak “*write less, do more / az yaz, çok iş yap*” cümlesini kullanan jQuery, özellikle sayfa içi etkileşim ve basit animasyonların gerçekleştirilebilmesini çok kolay hâle getiren bir JavaScript kütüphanesidir.

jQuery Nasıl Çalışır?



jQuery kütüphanesi, yerel bir adreste yer alabileceği gibi, Internet üzerindeki bir depodan da kullanılabilir.

JavaScript kütüphanelerinin genel çalışma prensipleri jQuery için de geçerlidir. Aslında basit bir metin dosyasından ibaret olan kütüphane dosyasına HTML kodunun yazıldığı dosyadan bağlantı verilmesi yeterli olmaktadır. jQuery kütüphane dosyasına iki farklı şekilde bağlantı sağlanabilir:

- Kütüphane dosyası yerel bir adreste bulunabilir (calıştığınız bilgisayarın sabit diskinde ya da sunucunuzun herhangi bir dizininde).
- Kütüphane dosyası jQuery'nin Internet üzerinde yer alan depolarından kullanılabilir.

Yukarıda belirttiğimiz bağlantı şekilleri aynı zamanda jQuery'nin nasıl kurulacağı ile ilgili durumu da belirler. jQuery kütüphane dosyasının *sürekli güncel kalması* isteniyorsa kütüphanenin Internet üzerindeki depodan kullanılması uygun olacaktır. Ancak eğer geliştirilecek uygulama Internet'e bağlı olmayan bir platform için ise (örneğin bir kiosk sistemi ya da bir gömülü sistem) kütüphane dosyasının yerel bir adreste bulunması gereklidir.

jQuery'nin nasıl çalıştığını gösteren kod bloku Şekil 12.1.'de gösterilmektedir. Şekil 12.1.'in 9. satırında yer alan adres tanımlaması kütüphane dosyasının yerel ya da Internet üzerindeki bulunmasına bağlı olarak değişiklik gösterebilir.

```
1. <!doctype html>
2. <html>
3. <head>
4.   <meta charset="utf-8">
5.   <title>Örnek Bağlantı</title>
6. </head>
7. <body>
8.   <a href="http://jquery.com/">jQuery</a>
9.   <script src="jquery.js"></script>
10.  <script>
11.    // jQuery kodları burada yer alacak!
12.
13.  </script>
14. </body>
15. </html>
```

Şekil 12.1. Jquery'nin Çalışma Şekli, Örnek Kod Bloku.

jQuery KURULUMU

jQuery ile ilgili kurulumda ihtiyaç duyacağımız tüm kaynak dosyalar <http://jquery.com/download/> adresinde yer almaktadır. Bu sayfada jQuery dosyalarının hem sıkıştırılmış hem de sıkıştırılmamış sürümleri yer almaktadır.

Sıkıştırılmamış dosya biçimini daha çok **kod üzerinde geliştirme çalışması** yapmak isteyenler için hazırlanmıştır; okunması, üzerinde değişiklik yapılması daha kolaydır. Sıkıştırılmış dosya biçiminin avantajı ise **boyut olarak daha küçük** olmasıdır. Bu şekilde HTML sayfalarının İnternet tarayıcıları tarafından yüklenmesi daha hızlı olur.



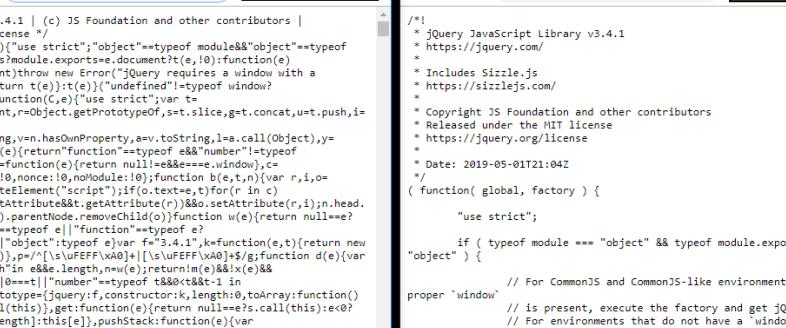
jQuery çekirdek dosyası
iki farklı biçimde
kullanılabilir:
sıkıştırılmış ya da
sıkıştırılmamış.

jQuery kullanımınız kütüphanenin hazır olan özelliklerini kullanmaktan ibaret olacaksız ve çekirdek kod üzerinde herhangi bir değişiklik yapmayacaksınız, her zaman sıkıştırılmış jQuery sürümlerini tercih etmeniz gereklidir. Bu şekilde çalışmanız Web sitesinin genel optimizasyonu açısından önem taşımaktadır.

jQuery dosya isimlendirmelerinde belirli bir algoritma kullanmaktadır. Bu sayede dosyanın adından ana sürüm bilgisi, alt sürüm bilgileri ve dosyanın biçimi hakkında bilgiler elde edilebilir. Aşağıdaki listede jQuery'nin son kararlı sürümü olan 3.4.1 için dosya isimlendirmeleri yer almaktadır:

- jQuery 3.4.1 geliştirme sürümü, sıkıştırılmış: jquery-3.4.1.min.js
 - jQuery 3.4.1 üretim sürümü, sıkıştırılmamış: jquery-3.4.1.js

Şekil 12.2.’de jQuery’nin çekirdek kodlarının sıkıştırılmış (sol) ve sıkıştırılmamış (sağ) dosya sürümleri karşılaştırma yapabilmeniz için yan yana yer almaktadır. Şekil 12.2.’de de görüldüğü gibi sıkıştırılmamış olan dosya biçimi **çok daha rahat** okunabilmektedir, ancak dosya boyutu açısından önemli bir fark bulunmaktadır. Özellikle yavaş internet bağlantılarında (mobil cihazlar vb. gibi) sayfanın yüklenme hızını etkileyebilir.



The image shows two side-by-side browser windows. Both windows have a yellow box highlighting the file size in the top right corner.

Left Window:

```
/*! jQuery v3.4.1 | (C) JS Foundation and other contributors |  
jquery.org/license */  
function(e,t){"use strict";"object"==typeof module&&"object"==typeof  
module.exports&module.exports=e.document?e:(t=10)=>function(e)  
(if(!e.documentElement)throw new Error("jQuery requires a window with a  
document object"));return t(e))("undefined"!=typeof window?  
window:document).Function("C","use strict";var t=/  
[1]=>C=document,_=obj=Object.getPrototypeOf,f=t.slice,g=t.concat,u=t.push,i=  
t.indexOf,n=t.lastIndexOf,o=String,vn=hasOwnProperty,a=vn.toString,l=a.call(Object),y=  
[],m=Function;if(!returnFunction="function"!=typeof e&&"number"!=typeof  
e.nodeType,y=e,x=function(e){return null==e&&e===window},c=  
{type:10,src:c,once:c,noModule:c},f=b,e,t,n=(var r,i=o=  
(n||!E).createElement("script"))if(o.text=e,t)for(r in c)  
i=[t].getattribute&t.getAttribute(r)&o.setAttribute(r,i);n.head.  
appendChild(o),parentNode.removeChild(o).function(e){return null==e?  
e:"object"==typeof e||"function"==typeof e?"object":e};  
n((o=Object.create(function(e){return f="1.4.1";function(e,t){return new  
fn.instantiate(e,t)};var fn={instantiate:e,parse:e,create:e,parseJSON:e,  
"array":e,join:e,"number":t&&t<1?1:  
e,k:fn.prototype}(jQuery,f,constructor:k,length:t&&t>1?Array:function()  
(return s).call(this)),get:fn.get||(e){return null==e?s.call(this):e<0?  
this[e].length:0},pushStack:fn.pushStack||(e){var  
t,k,merge:(this,constructor),e};return  
t.prevObject=t.jstxt,each:fn.each||(e){return  
k.each(e,fn.map||(n){return  
this.pushStack(k=this,fn.map||(e){return  
this.pushStack(k,fn(this,e),e)}(e))}))});
```

Right Window:

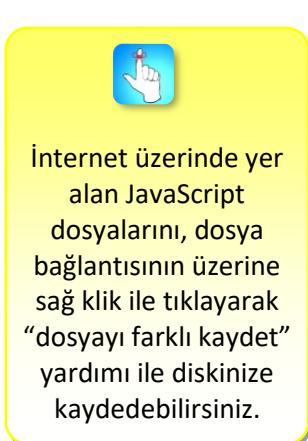
```
/*!  
* jQuery JavaScript Library v3.4.1  
* https://jquery.com/  
*  
* Includes Sizzle.js  
* https://sizzlejs.com/  
*  
* Copyright JS Foundation and other contributors  
* Released under the MIT license  
* https://jquery.org/license  
*  
* Date: 2019-05-01T21:04Z  
*/  
(function(global, factory) {  
    "use strict";  
  
    if ( typeof module === "object" && typeof module.exports ===  
        "object" ) {  
  
        // For CommonJS and CommonJS-like environments where a  
        // proper 'window'  
        // is present, execute the factory and get jQuery.  
        // For environments that do not have a 'window' with a  
        'document'  
        // (such as Node.js), expose a factory as  
        module.exports.  
        // This accentuates the need for the creation of a real
```

Şekil 12.2. Jquery Sıkıştırılmış (Sol) Ve Sıkıştırılamamış (Sağ) Dosya Biçimleri Ve Boyutları.

Şimdi jQuery'nin sıkıştırılmış dosya biçimindeki kütüphane dosyasını (`jquery-3.4.1.js`) sabit diskimizde oluşturduğumuz bir proje klasörü içerisine <http://jquery.com/download/> adresinden yükleyerek kaydedelim. Kaydetme işlemi sırasında kütüphane dosyasına istediğiniz bir ismi verebilir ya da mevcut ismi değiştirmeden kullanabilirsiniz. Bu çalışmada dosya ismimizi "jquery.js" olarak sadeleştiriyoruz.

Şekil 12.3.’te <http://jquery.com/download/> adresinde jQuery başlığı altında yer alan “*Download the compressed, production jQuery 3.4.1*” bağlantısı

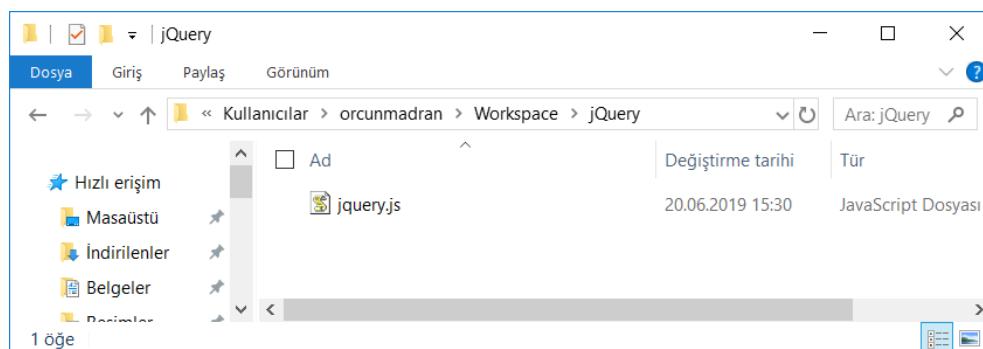
görmektedir. Bu bağlantıya sağ tıklayarak “**bağlantıyı farklı kaydet**” seçeneği ile dosyayı yerel diskinize oluşturduğunuz “jQuery” proje klasörünün içine kaydediniz.



The screenshot shows the official jQuery download page. At the top, there's a message about upgrading: "For help when upgrading jQuery, please see the [upgrade guide](#) most relevant to your version. We also recommend using the [jQuery Migrate plugin](#)". Below this, three download links are provided: "Download the compressed, production jQuery 3.4.1", "Download the uncompressed, development jQuery 3.4.1", and "Download the map file for jQuery 3.4.1".

Şekil 12.3. Jquery Yükleme Sayfası Ve Kütüphane Dosyası Bağlantısı.

Kayıt işlemini gerçekleştirdikten sonra jQuery adlı proje klasörümüzün görünümü Şekil 12.4.'teki gibi olmalıdır.



Şekil 12.4. Sabit Diskimizde Yer Alan “Jquery” Adlı Proje Klasörü.

Kurulum işleminin dosya yapılandırma aşamasını tamamladığımıza göre artık test için bir HTML dokümanı oluşturup jQuery kütüphane dosyası ile bağlantısını kurabiliriz. Oluşturacağımız HTML dokümanının ismi “test.html” ve içeriği de Şekil 12.5.'teki gibi olsun.

```
1. <!doctype html>
2. <html>
3. <head>
4.   <meta charset="utf-8">
5.   <title>jQuery Test</title>
6.   <script src="jquery.js"></script>
7.   <script>
8.     $(document).ready(function() {
9.       $("p").click(function() {
10.         $(this).hide();
11.       });
12.     });
13.   </script>
```

```
14. </head>
15. <body>
16.     <p>Tıkla ve gizle..!</p>
17. </body>
18. </html>
```

Şekil 12.5. Jquery Test Dokümanı İçinde Yer Alan Kod Bloku.

Şekil 12.5.'teki kodu incelediğimizde başlık (<head>) bölümü içerisinde 6. satırda jQuery kütüphanesi ile bağlantı kurulduğunu görüyoruz. 7. ve 13. satırlar arasında ise JavaScript kodlarımız yer almaktadır. Bu test çalışmasında sayfanın gövde bölümü (<body>) içerisinde yer alan <p> etiketi ile girilmiş metinlerin üzerine tıklandığı zaman metnin gizlenmesini (bir anlamda sayfadan kaybolmasını) amaçlıyoruz.

Eğer yukarıdaki adımları doğru bir şekilde yaptıysanız, test.html dosyasını herhangi bir internet tarayıcısında açtığınızda “Tıkla ve gizle..!” metnine tıklayıp yok olduğunu görebilmeniz gereklidir.



İçerik Dağıtım Ağı (CDN) sistemi ile jQuery kütüphanelerini sabit diskinize kaydetmeden, internet üzerinden de kullanabilirsiniz.

Bu örneğimizi jQuery kütüphane dosyasına internet üzerindeki bir kaynaktan bağlantı sağlayarak da test edebiliriz. Bu sayede ilgili kütüphane sürümünün *her zaman en güncel* hâlini kullanmış oluruz. Şekil 12.5.'te yer alan test kodumuzun 6. satırını aşağıdaki gibi değiştirelim. Bu değişikliği yaptıktan sonra test.html dosyasını tarayıcıda açalım. internet bağlantımızın olması gerektiğini de unutmayalım.

```
6. <script src="http://code.jquery.com/jquery-
3.4.1.min.js"></script>
```

CDN (Content Delivery Network) yani İçerik Dağıtım Ağı adı verilen bu sistem ile birçok farklı kaynak kod sabit diske kaydedilmeden de kullanılabilir. jQuery'nin CDN sistemine <http://code.jquery.com/> adresinden ulaşılabilir ve farklı sürümleri bu sayfa aracılığıyla kullanabilirsiniz.

jQuery ile ilgili kurulum aşamaları tamamlandı. Artık jQuery ile ilgili daha detaylı çalışmaları yapabilmek için gerekli altyapıya sahipsiniz.



Bireysel Etkinlik

- Sabit diskinizde "jQuery" adlı bir proje klasörü oluşturun.
- Proje klasörünüzüne içine jQuery kütüphane dosyasını "jquery.js" olarak Şekil 12.3.'teki bağlantıyı kullanarak Şekil 12.4.'teki gibi kaydedin.
- Şekil 12.5.'te yer alan kodları test.html dosyasına yerleştirerek internet tarayıcınızda görüntüleyin.

jQuery SÖZDİZİMİ (Syntax)

jQuery sözdizimi HTML öğelerinin (etiketlerinin) ve CSS tanımlamalarının

seçilmesi ve bu ögelere birtakım eylemler atanması şeklinde gerçekleştirilir.



jQuery, sayfa henüz tam olarak yüklenmeden kodların çalışmaya başlamasını engellemek için **doküman hazır olayı** adında bir fonksiyon kullanır.



Örnek

- Sözdizimi şablonu: `$(seçici).eylem()`
- Mevcut ögenin gizlenmesi: `$(this).hide()`
- `<p>` etiketli tüm öğelerin gizlenmesi: `$("p").hide()`
- "test" sınıfına dâhil tüm öğelerin gizlenmesi: `$(".test").hide()`
- "test" kimliğine sahip tüm öğelerin gizlenmesi: `$("#test").hide()`

jQuery sözdizimi içerisinde önemli noktalardan biri de "The Document Ready Event" adı verilen **doküman hazır olayı**dır. Bu olay genel kullanımda jQuery kodlarını kapsayıcı bir fonksiyondur. Kullanılan bu fonksiyon sayesinde dokümanın yüklenmesi henüz tamamlanmadan jQuery kodlarının çalışması engellenir ve olası bir hatadan kaçınılmış olur.

Doküman hazır olayının kullanımı Şekil 12.6.'da gösterilmiştir. Rahat okunması amacıyla çoklu satır olarak görüntülenen bu kod bloku aslında tek satır olarak yorumlanır.

```
1. $(document).ready(function () {  
    // jQuery kodları bu bölümde yer alacak.  
});
```

Şekil 12.6. Doküman Hazır Oayı Örnek Kodları.

jQuery geliştirici takımı bu olay için daha kısa bir metot da oluşturmuştur. Kısa metot Şekil 12.7.'de gösterilmektedir.

```
$(function () {  
    // jQuery kodları bu bölümde yer alacak.  
});
```

Şekil 12.7. Doküman Hazır Oayı Kısıtlımsız Sürümü.

Şekil 12.6. ve Şekil 12.7.'deki kod bloklarından hangisini tercih edeceğiniz size kalmıştır. Genel olarak tavsiye edilen ve bu ünite içerisindeki kullanılan metot Şekil 12.6.'daki metot olacaktır.

jQuery Seçicileri (Selectors)

jQuery seçicileri HTML ögelerinin seçimini ve bu ögeler ile ilgili değişiklikleri mümkün kılar. Seçiciler HTML ögelerini adlarına, kimliklerine (id), sınıflarına (class), tiplerine, niteliklerine, niteliklerinin sahip olduğu değerlere göre seçebilirler.

Daha önce test dokümanımızda örneğini gördüğümüz gizleme olayını ele alalım. Bu olay sayfadaki herhangi bir HTML ögesini gizlememize olanak sağlar. HTML ögesi olarak paragraf tanımlayıcısı olan “p” etiketini seçelim. Örnek kod Şekil 12.8.’deki gibi olmalıdır.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.     $("button").click(function() {
8.         $("p").hide();
9.     });
10. });
11.</script>
12.</head>
13.<body>
14.
15.<h2>jQuery' e Hoşgeldiniz</h2>
16.
17.<p>jQuery bir JavaScript (JS) kütüphanesidir.<p>
18.<p>Hızlı ve kolay JS kodlama yapmanızı sağlar!</p>
19.
20.<button>Paragrafları gizlemek için tıklayınız</button>
21.
22.</body>
23.</html>
```



jQuery seçicileri, aynen CSS sözdiziminde olduğu gibi HTML öğelerini etiketleri, sınıfları ya da kimlikleri ile seçebilirler.

Şekil 12.8. Jquery Seçici Örnek Kodu.

Şekil 12.8.’deki kodu incelediğimizde 6. satırda öncelikli olarak tüm sayfanın yüklenmesini bekleyen fonksiyonu görüyoruz. Bu fonksiyondan sonra 8. satırda HTML ögesi olarak “p” seçilmiş durumda; sayfa içerisinde <p> etiketi ile tanımlanmış olan tüm metinleri etkileyeyecek bir seçim.

Eğer 8. satırda yer alan seçiciyi “p” yerine “h2” olarak değiştirirsek, bu sefer gizlenecek olan öge ikinci seviyeden başlık olan <h2> etiketi ile tanımlanmış metin olacaktır.



Bireysel Etkinlik

- Proje klasörünüzün içinde "secici.html" adlı bir dosya oluşturun.
- Şekil 12.8.'de yer alan kodları secici.html dosyasına yerleştirerek Internet tarayıcıda görüntüleyin.
- "secici.html" içinde yer alan kodlarda jQuery seçicisini <h2> etiketini seçerek şekilde yeniden düzenleyin ve test edin.

jQuery'nin birçok farklı şekilde seçim işlemini gerçekleştirebileceğini belirtmiştim. Bir HTML sayfası içinde yer alan tüm paragrafları değil, sadece belirlediğimiz paragrafları gizlemek istiyoruz. Bu noktada bir HTML etiketi değil, etiketin içindeki özel bir sınıfı seçim işlemi için kullanabiliriz. Örnek uygulama Şekil 12.9.'daki gibi kodlanmalıdır.



Bir HTML sayfası içerisinde belirli bir öğe grubunun içinde sınıfları kullanarak özel seçimler yapabilirsiniz.

```

1.  <!DOCTYPE html>
2.  <html>
3.  <head>
4.  <script src="jquery.js"></script>
5.  <script>
6.      $(document).ready(function() {
7.          $("button").click(function() {
8.              $(".ilk").hide();
9.          });
10.     });
11.    </script>
12.    </head>
13.    <body>
14.        <h2>jQuery'e Hoşgeldiniz</h2>
15.        <p class="ilk">jQuery bir JavaScript (JS)
kütüphanesidir.<p>
16.        <p>Hızlı ve kolay JS kodlama yapmanızı sağlar!</p>
17.        <button>İlk paragrafi gizlemek için
tıklayınız</button>
18.    </body>
19.  </html>

```

Şekil 12.9. Jquery Sınıf Seçimi Örnek Kodları.

jQuery Olayları

jQuery olayları, HTML sayfası içindeki etkileşimlere karşılık vermemizi mümkün kılar. Bu etkileşimler farenin bir HTML ögesi üzerinden geçmesi, bir form elemanın seçili hâle getirilmesi ya da HTML sayfası içerisindeki herhangi bir öğeye tıklanması şeklinde ortaya çıkabilir. Tablo 12.1.'de Web sayfalarında sıkça kullanılan olayların jQuery içerisinde kullanılan kodları yer almaktadır.

Tablo 12.1. Web Sayfalarında Sıkça Kullanılan Olaylar.

Fare Olayları	Klavye Olayları	Form Olayları	Doküman Olayları
click (tıklama)	keypress (tuşa basma)	submit (gonderme)	load (yükleme)
dblclick (çift tıklama)	keydown (tuş basılı)	change (değişime)	size (boyutlandırma)
mouseenter (fare girişi)	keyup (tuşu bırakma)	focus (odaklanma)	scroll (kaydırma)
mouseleave (fare çıkışı)		blur (bulanıklaşma)	unload (çıkma)

Bir sayfadaki paragraflara tıklandığı zaman herhangi bir fonksiyonun tetiklenmesi isteniyorsa kullanılacak kod yapısı Şekil 12.10.'daki gibi olmalıdır.

```

$( "p" ).click(function() {

    // tetiklenecek kodlar burada yer alacak

});

```

Şekil 12.10. Jquery Olay Yapısı.

jQuery'deki olay yapısını daha iyi anlayabilmek için bir örnek üzerinde çalışalım. Örnek uygulamamızda Web sayfası içerisinde yer alan paragrafların üzerine fare ile geldiğimizde bize her paragraf için farklı bir uyarı versin. Bu çalışmanın kodları Şekil 12.11.'deki gibi olmalıdır.



Bir olay birden fazla fonksiyonu tetikleyebilir.

```

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.     $("#p1").mouseenter(function() {
8.         alert("1. paragrafa giriş yaptınız!");
9.     });
10.    $("#p2").mouseenter(function() {
11.        alert("2. paragrafa giriş yaptınız!");
12.    });
13. });
14. </script>
15. </head>
16. <body>
17. <p id="p1">Fare ile bu paragrafın üzerine gelin.</p>
18. <p id="p2">Fare ile 2. paragrafın üzerine gelin.</p>
19. </body>
20. </html>
```

Şekil 12.11. Jquery Olay Yapısı Örnek Uygulama.

Şekil 12.11.'deki kodları incelediğinizde daha önce Tablo 12.1.'de fare olayı olarak listelediğimiz "mouseenter" (fare girişi) olayın kullanımını göreceksiniz. Bu olay ilgili paragrafin üzerine fare ile gelindiğinde bir uyarı kutusunun görüntülenmesini tetiklemektedir. 7. ve 10. satılardaki "mouseenter" olayı farklı bir olay ile değiştirildiğinde (örneğin "click") olayın tetiklenme şekli de tahmin edebileceğiniz gibi değişecektir.



Bireysel Etkinlik

- Proje klasörünüzün içinde "olay.html" adlı bir dosya oluşturun.
- Şekil 12.11.'de yer alan kod yapısını örnek alarak sayfada yer alan bir resim ögesinin üzerine çift tıklandığında bir uyarı mesajı olmasını sağlayacak kodları olay.html dosyası içinde yazın ve test edin.

jQuery EFEKTLERİ

jQuery'nin kullanıcıları en çok etkileyen özellikleri HTML öğeleri ile yapılan efektlerdir. Bu efektler sayesinde sayfada yer alan öğeler gizlenebilir ya da gösterilebilir. Görsellere kaydırma efekti uygulanarak slayt gösterileri (günümüz Web sitelerinin neredeyse temel özelliklerinden biri hâline gelmiştir) düzenlenebilir. Web sayfalarında sıkılıkla kullanılan jQuery efektleri aşağıda listelenmiştir:

- Gizle / Göster (Show / Hide)
- Soldurma (Fade)
- Kaydırma (Slide)
- Hareketlendirme (Animate)



jQuery efektlerini kullanarak etkileyici Web sayfaları oluşturabilirsiniz.

- Durdurma (Stop)

Gizle / Göster (Show / Hide)

jQuery ile HTML elemanlarını hide() ve show() metodları ile gizleyebilir ya da görünür hâle getirebilirsiniz. hide() ve show() metodlarının sözdizimi Şekil 12.12.'de görüldüğü gibidir.

```
1. $(seçici).hide(gizleme hızı);  
2. $(seçici).show(gösterme hızı);
```

Şekil 12.12. Hide() Ve Show() Metotlarının Sözdizimi

hide() ve show() metodlarının kullanımını için bir örnek bir uygulama yapalım. Örnek uygulamamızda sayfa içerisindeki bir paragrafi “gizle” butonu ile gizleyip, “göster” butonu ile görünür hâle getirelim. Bu çalışmanın kodları şekil 12.13.'teki gibi olmalıdır.

```
1. <!DOCTYPE html>  
2. <html>  
3. <head>  
4. <script src="jquery.js"></script>  
5. <script>  
6. $(document).ready(function () {  
7.     $("#hide").click(function () {  
8.         $("p").hide(1000);  
9.     });  
10.    $("#show").click(function () {  
11.        $("p").show(1000);  
12.    });  
13.});  
14. </script>  
15. </head>  
16. <body>  
17. <p>jQuery ile paragraf gizle, göster!</p>  
18. <button id="hide">Gizle</button>  
19. <button id="show">Göster</button>  
20. </body>  
21. </html>
```



jQuery efektlerinin
gerçekleşme hızı
fonksiyonlar içinde
milisaniye olarak
tanımlanabilmektedir.

Şekil 12.13. Hide() Ve Show() Örnek Uygulama.

Örnek uygulamamızın 8. ve 11. satırlarında yer alan “hide” ve “show” fonksiyonlarının içinde rakam ile 1000 tanımlaması yapıldığı gözünüzden kaçmamıştır. Bu tanımlama gizleme ve gösterme işleminin 1 saniyede (1000 milisaniye) gerçekleşmesi komutunu fonksiyona göndermektedir. Fonksiyona herhangi bir değer gönderilmemesi durumunda işlem anında gerçekleşmektedir.



Bireysel
Etkinlik

- Proje klasörünüzün içinde "gizleGoster.html" adlı bir dosya oluşturun.
- Şekil 12.13.'te yer alan kod yapısını örnek alarak sayfada yer alan bir paragrafi butonlar yardımıyla gizlemeye ve göstermeye çalışın.

jQuery efektleri için kullanılan metodlarda "**Toggle**" adlı özel bir metot türü bulunmaktadır. Bu metot türü ile tek bir tetikleyici kullanarak (örneğin Web sayfasında yer alan tek bir buton yardımıyla) işlemin ilk tetiklemede (fare üzerinde tıklamada) gerçekleşip, ikinci tetiklemede ise (fare üzerinde yeniden tıklamada) işlemin geri alınması sağlanabilmektedir. Şekil 12.14.'te Web sayfasında yer alan butona ilk tıklanışta paragraf gizlenecek, aynı butona tekrar basıldığında ise paragraf görünür hâle gelecektir.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.     $("button").click(function() {
8.         $("p").toggle();
9.     });
10. });
11. </script>
12. </head>
13. <body>
14. <button>Hem gizle, hem göster!</button>
15. <p>Bu paragraf butona ilk tıklandığında gizlenecek,
   butona ikinci tıklandığında görünür hale gelecek</p>
16. </body>
17. </html>
```

Şekil 12.14. Jquery Metotlarında “Toggle” Kullanımı.



Bireysel
Etkinlik

- Proje klasörünüzün içinde "toogle.html" adlı bir dosya oluşturun.
- Şekil 12.14.'te yer alan kod yapısını örnek alarak sayfada yer alan bir paragrafi tek bir buton yardımıyla gizlemeye ve göstermeye çalışın.

Soldurma (Fade)

jQuery ile HTML elemanlarını “fade” metodunu kullanarak soldurma efekti ile gizleyebilir (fadeOut) ya da belirginleştirme efekti ile görünür hâle (fadeIn) getirebilirsiniz. jQuery içerisinde “fade” metodunun farklı kullanım şekilleri yer almaktadır. Bunlar;

- *fadeIn()*: Belirginleştirerek gösterme,
- *fadeOut()*: Soldurarak gizleme,
- *fadeToggle()*: Soldurarak gizleme ve belirginleştirerek gösterme,
- *fadeTo()*: Öğeyi soldururken belirli bir saydamlık değeri verme.

“fade” metodlarının sözdizimleri Şekil 12.15.’te görüldüğü gibidir.

```
1. $(seçici).fadeIn(hız);  
2. $(seçici).fadeOut(hız);  
3. $(seçici).fadeToggle(hız);  
4. $(seçici).fadeTo(hız,saydamlık);
```

Şekil 12.15. “Fade” Metotlarının Sözdizimi

fadeIn() fonksiyonu

fadeIn() fonksiyonunun örnek uygulaması Şekil 12.16.’da görüldüğü gibidir. Örnek uygulamada, Web sayfasında gizlenmiş şekilde bulunan kırmızı, yeşil ve mavi renkli kutular farklı efekt parametreleri ile görünür hâle gelmektedir.



jQuery'de efektlerin
gerçekleşme hızını
belirlerken “fast” (hızlı)
ve “slow” (yavaş)
anahtar kelimelerini de
kullanabilirsiniz.

```
1. <!DOCTYPE html>  
2. <html>  
3. <head>  
4. <script src="jquery.js"></script>  
5. <script>  
6. $(document).ready(function() {  
7.     $("button").click(function() {  
8.         $("#div1").fadeIn();  
9.         $("#div2").fadeIn("slow");  
10.        $("#div3").fadeIn(3000);  
11.    });  
12.});  
13. </script>  
14. </head>  
15. <body>  
16. <h1>fadeIn fonksiyonu örnek uygulaması</h1>  
17. <button>Kutuları görüntülemek için  
tıklayınız!</button><br><br>  
18. <div id="div1"  
style="width:80px;height:80px;display:none;background-  
color:red;"></div><br>  
19. <div id="div2"  
style="width:80px;height:80px;display:none;background-  
color:green;"></div><br>  
20. <div id="div3"  
style="width:80px;height:80px;display:none;background-  
color:blue;"></div>  
21. </body>  
22. </html>
```

Şekil 12.16. *FadeIn()* Fonksiyonu Örnek Uygulaması.

fadeOut() fonksiyonu

fadeOut() fonksiyonunun örnek uygulaması Şekil 12.17.’de görüldüğü gibidir. Örnek uygulamada, Web sayfasında yer alan kırmızı, yeşil ve mavi renkli

kutular farklı efekt parametreleri ile gizlenmektedirler.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.     $("button").click(function() {
8.         $("#div1").fadeOut();
9.         $("#div2").fadeOut("slow");
10.        $("#div3").fadeOut(3000);
11.    });
12. });
13. </script>
14. </head>
15. <body>
16. <h1>fadeOut() fonksiyonu örnek uygulaması.</h1>
17. <button>Kutuları gizlemek için  
tıklayınız!</button><br><br>
18. <div id="div1"  
style="width:80px;height:80px;background-  
color:red;"></div><br>
19. <div id="div2"  
style="width:80px;height:80px;background-  
color:green;"></div><br>
20. <div id="div3"  
style="width:80px;height:80px;background-  
color:blue;"></div>
21. </body>
22. </html>
```

Şekil 12.17. Fadeout() Fonksiyonu Örnek Uygulaması.

fadeToggle() fonksiyonu

fadeToggle() fonksiyonunun örnek uygulaması Şekil 12.18.'de görüldüğü gibidir. Örnek uygulamada, butona ilk tıkladığında Web sayfasında yer alan kırmızı, yeşil ve mavi renkli kutular gizlenmekte, ikinci kere tıklandığında ise kutular görüntülenmektedir.



jQuery efektleri tüm
HTML elemanlarına
uygulanabilir.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.     $("button").click(function() {
8.         $("#div1").fadeToggle();
9.         $("#div2").fadeToggle("slow");
10.        $("#div3").fadeToggle(3000);
11.    });
12. });
13. </script>
14. </head>
15. <body>
16. <h1>fadeToggle() fonksiyonu örnek uygulaması</h1>
```

```
17. <button>Kutuları gizleyip, göstermek iççin  
tıklayınız!</button><br><br>  
18. <div id="div1"  
style="width:80px; height:80px; background-  
color:red;"></div>  
19. <br>  
20. <div id="div2"  
style="width:80px; height:80px; background-  
color:green;"></div>  
21. <br>  
22. <div id="div3"  
style="width:80px; height:80px; background-  
color:blue;"></div>  
23. </body>  
24. </html>
```

Şekil 12.18. Fadetoggle() Fonksiyonu Örnek Uygulaması.

fadeTo() fonksiyonu



Saydamlık (arka planı gösterme) jQuery efektlerin içinde kullanılabilir parametrelerden biridir.

fadeTo() fonksiyonunun örnek uygulaması Şekil 12.19.'da görüldüğü gibidir. Bu çalışmada, butona tıklandığında Web sayfasında yer alan kırmızı, yeşil ve mavi renkli kutulara farklı saydamlık parametreleri uygulanmaktadır.

```
1. <!DOCTYPE html>  
2. <html>  
3. <head>  
4. <script src="jquery.js"></script>  
5. <script>  
6. $(document).ready(function() {  
7.     $("button").click(function() {  
8.         $("#div1").fadeTo("slow", 0.15);  
9.         $("#div2").fadeTo("slow", 0.4);  
10.        $("#div3").fadeTo("slow", 0.7);  
11.    });  
12.});  
13. </script>  
14. </head>  
15. <body>  
16. <h1>fadeTo() fonksiyonu örnek uygulaması.</h1>  
17. <button>Kutuları saydam hale getir!</button><br><br>  
18. <div id="div1"  
style="width:80px; height:80px; background-  
color:red;"></div><br>  
19. <div id="div2"  
style="width:80px; height:80px; background-  
color:green;"></div><br>  
20. <div id="div3"  
style="width:80px; height:80px; background-  
color:blue;"></div>  
21. </body>  
22. </html>
```

Şekil 12.19. Fadeto() Fonksiyonu Örnek Uygulaması.

Şekil 12.19.'un 8., 9. ve 10. satırlarında ondalık değerler saydamlığı belirleyen numerik değerlerdir. Değer 1'e yaklaşıkça saydamlık oranı düşer. Örnekteki değerler göz önüne alındığında, kırmızı kutu saydamlığı en yüksek olan kutudur.



Bireysel Etkinlik

- Proje klasörünüzün içinde "soldurma.html" adlı bir dosya oluşturun.
- Şekil 12.19.'da yer alan kod yapısını örnek alarak sayfada oluşturduğunuz kutuların saydamlık değerlerini bir buton yardımıyla değiştirmeye çalışın.

Kayma (Slide)



jQuery efektleri içerisinde kullanılan "Toggle" metodu yardımıyla tek bir buton ile iki farklı olay gerçekleştirilebilir.

jQuery ile HTML elemanlarını "slide" metodunu kullanarak kaydırabilirsiniz. jQuery içerisinde "slide" metodunun farklı kullanım şekilleri yer almaktadır. Bunlar;

- *slideDown()*: Aşağıya doğru kaydırma,
- *slideUp()*: Yukarı doğru kaydırma,
- *slideToggle()*: Aşağıya daha sonra ise yukarıya kaydırma (ya da tam tersi).

"slide" metodlarının sözdizimleri Şekil 12.20.'de görüldüğü gibidir.

```
1. $(seçici).slideDown(hız);
2. $(seçici).slideUp(hız);
3. $(seçici).slideToggle(hız);
```

Şekil 12.20. "Slide" Metotlarının Sözdizimi

slideDown() fonksiyonu

slideDown() fonksiyonunun örnek uygulaması Şekil 12.21.'de görüldüğü gibidir. Örnek uygulamada, HTML elemanı olarak oluşturulmuş olan bir panel aşağıya doğru kayarak açılmaktadır.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.   $("#flip").click(function() {
8.     $("#panel").slideDown("slow");
9.   });
10. });
11. </script>
12. <style>
13. #panel, #flip {
14.   padding: 5px;
15.   text-align: center;
16.   background-color: #e5eecc;
17.   border: solid 1px #c3c3c3;
18. }
19. #panel {
20.   padding: 50px;
21.   display: none;
22. }
23. </style>
```

```
24. </head>
25. <body>
26. <div id="flip">Panel'i açmak için tıklayınız!</div>
27. <div id="panel">jQuery kayan panel örneği...</div>
28. </body>
29. </html>
```

Şekil 12.21. Slidedown() Fonksiyonu Örnek Uygulaması.



jQuery efektlerinin daha hızlı gerçekleşmesini istiyorsanız kod içindeki "slow" değerini "fast" ile değiştirebilirsiniz.

slideUp() fonksiyonu

slideUp() fonksiyonunun örnek uygulaması Şekil 12.22.'de görüldüğü gibidir. Örnek uygulamada, HTML elemanı olarak oluşturulmuş olan bir panel yukarıya doğru kayarak kapanmaktadır.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.   $("#flip").click(function() {
8.     $("#panel").slideUp("slow");
9.   });
10. });
11. </script>
12. <style>
13. #panel, #flip {
14.   padding: 5px;
15.   text-align: center;
16.   background-color: #e5eecc;
17.   border: solid 1px #c3c3c3;
18. }
19. #panel {
20.   padding: 50px;
21. }
22. </style>
23. </head>
24. <body>
25. <div id="flip">Panel'i kapatmak için
26.   tıklayınız!</div>
27. <div id="panel">jQuery kayan panel örneği...</div>
28. </body>
29. </html>
```

Şekil 12.22. Slidedown() Fonksiyonu Örnek Uygulaması.

Örnek uygulamanın yer aldığı Şekil 12.22.'nin 12. ve 22. satırları arasında yer alan bölüm CSS ile panelin tanımlandığı bölümdür. Bu bölüm üzerinde değişiklikler yapılarak farklı renk ve boyutta paneller elde edilebilir.

slideToggle() fonksiyonu

slideToggle() fonksiyonunun örnek uygulaması Şekil 12.23.'te görüldüğü gibidir. Örnek uygulamada, butona ilk tıkladığında kayarak açılan panel butona tekrar tıklandığında kayarak kapanmaktadır.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.   $("#flip").click(function() {
8.     $("#panel").slideToggle("slow");
9.   });
10. });
11. </script>
12. <style>
13. #panel, #flip {
14.   padding: 5px;
15.   text-align: center;
16.   background-color: #e5eecc;
17.   border: solid 1px #c3c3c3;
18. }
19. #panel {
20.   padding: 50px;
21.   display: none;
22. }
23. </style>
24. </head>
25. <body>
26. <div id="flip">Paneli açmak ya da kapatmak için  
tıklayınız!</div>
27. <div id="panel">jQuery kayan panel örneği...</div>
28. </body>
29. </html>
```



Farklı HTML elemanları ile çalışmak için ilgili elemanın CSS özelliklerini değiştirebilirsiniz.



- Bireysel Etkinlik
- Proje klasörünüzün içinde "panel.html" adlı bir dosya oluşturun.
 - Şekil 12.23.'te yer alan kod yapısını örnek alarak sayfa içerisinde açılıp kapanabilen bir bilgi paneli oluşturmaya çalışın.

Hareketlendirme (Animate)

jQuery ile HTML elemanlarına kendi belirlediğiniz şekilde (önceden tanımlanmış efektlerden bağımsız olarak) farklı hareket ve efektler vererek animasyonlar oluşturabilirsiniz. Bu animasyonlar HTML elemanlarının sayfa

üzerindeki konumlarını değiştirek, renk ve boyutları üzerinde değişiklikler yaparak gerçekleştirilmektedir. jQuery'deki hareketlendirme uygulamalarını daha iyi anlayabilmek için animate() fonksiyonunun işleyişini genel hatları ile inceleyelim. animate() fonksiyonunun sözdizimi Şekil 12.24.'teki gibidir.

```
1. $(seçici).animate({parametreler}, hız);
```

Şekil 12.24. Animate() Fonksiyonu Sözdizimi.

Şekil 12.24.'teki sözdizimini bir örnek uygulamada kullanalım. Örnek uygulamamızda Web sayfasında yer alan yeşil renkli bir kutu bulunduğu konumdan itibaren 250 piksel sağa doğru hareket etsin. Bu çalışmadaki kod yapısı Şekil 12.25.'teki gibi olmalıdır.



jQuery animate()
fonksiyonu ile kendi
belirlediğiniz
animasyonları
oluşturabilirsiniz.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.   $("button").click(function() {
8.     $("div").animate({left: '250px'});
9.   });
10. });
11. </script>
12. </head>
13. <body>
14. <button>Animasyona başla!</button>
15. <p>Varsayılan olarak tüm HTML elemanları sayfa
   içerisinde static bir konumdadır. CSS'in konum özelliği
   sayesinde sayfa içindeki konumları değiştirebilir.</p>
16. <div
   style="background:#98bf21; height:100px; width:100px; posi
   tion:absolute;"></div>
17. </body>
18. </html>
```

Şekil 12.25. Animate() Fonksiyonu Örnek Uygulama.

Şekil 12.25.'teki örnek uygulamanın 8. satırında animate() fonksiyonu içerisinde bazı parametrelerin tanımlandığını görmüşsunuzdur. Bu parametreler CSS'in konum özellikleri kullanarak HTML elemanın konumunu değiştirmektedir. Yeşil karenin sağa doğru hareket etmesine rağmen neden "left" yani sol olarak parametre tanımladığı sorulabilir. Bunun cevabı soldan itibaren 250 piksel hareket etmesi istediği içindir. CSS içerisindeki tanımlamalar bu şekilde yapılır.



jQuery animate()
fonksiyonu ile HTML
elemanlarının birden
çok özelliği
değiştirilebilir.

jQuery'deki animasyon özelliği sadece tek bir özelliğin değiştirilmesi ile sınırlı değildir. HTML elemanın birden çok özelliği de değiştirilebilir ve animasyon daha karmaşık bir yapıya kavuşabilir. Şekil 12.26.'da önceki örneğimizde kullandığımız yeşil kare bu sefer sağa doğru 300 piksel hareket etmekte ve boyutlar ile saydamlık değeri de değişiklik göstermektedir.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.     $("button").click(function() {
8.         $("div").animate({
9.             left: '250px',
10.            opacity: '0.5',
11.            height: '150px',
12.            width: '150px'
13.        });
14.    });
15. });
16. </script>
17. </head>
18. <body>
19. <button>Animasyona başla!</button>
20. <p>Varsayılan olarak tüm HTML elemanları sayfa
   içerisinde statik bir konumdadır ve boyut ve renk gibi
   özellikleri de sabittir. CSS sayesinde belirlenen
   özellikler sayfa içinde değişimdir.</p>
21. <div
   style="background:#98bf21;height:100px;width:100px;posi
   tion:absolute;"></div>
22. </body>
23. </html>
```

Şekil 12.26. Animate() Fonksiyonu İle Birden Fazla Özelliğin Değiştirilmesi.

animate() fonksiyonunu kullandığımız örneklerimizde (Şekil 12.24., 16. satır, Şekil 12.26., 21. satır) HTML elemanın (yeşil kutunun) ilk tanımlaması `<div>` etiketleri yardımıyla yapılır. Bu tanımlamaları istediğiniz gibi değiştirerek farklı HTML elemanları oluşturabilirsiniz.

jQuery'deki animasyon kullanımının ilginç noktalarından biri de HTML elemanın CSS özelliklerini değiştirirken verilen değerin sabit değil, bir önceki değere bağlı olarak verilebilmesidir. Bu özelliği gözlemleyebileceğimiz örneğimizin kodları Şekil 12.27.'de yer almaktadır. Örnek uygulamada sayfada yer alan kırmızı bir kutu, butona her tıklandığında 50 piksel sağa doğru hareket edecektir.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.     $("button").click(function() {
8.         $("div").animate({
9.             left: '+=50px'
10.        });
11.    });
12. });
13. </script>
14. </head>
15. <body>
```

```
16. <button>Animasyona başla!</button>
17. <p>Bir önceki konumuna göre hareketini güncelleyen
   kutu örneği.</p>
18. <div
   style="background:#FF0000;height:100px;width:100px;posi
   tion:absolute;"></div>
19. </body>
20. </html>
```

Şekil 12.27. Önceki Değere Bağlı Olarak Animasyonun Gerçekleşmesi.

Şekil 12.27.'deki örneğin 9. satırında parametre tanımlaması yapılırken statik bir değer belirlemek yerine “+=” ifadesi kullanılarak bir önceki konuma 50 piksel eklenmesi sağlanmaktadır. Bu sayede **butona her tıklandığında** kırmızı kutunun mevcut konumu değişmektedir.



Tek bir fare hareketi ile
birden çok animasyon
tetiklenebilir.

jQuery'deki animate() fonksiyonu tek bir tetikleme ile birden çok animasyonun arka arkaya gerçekleştirilemesine de imkân sağlar. Şekil 12.28.'de mavi bir kutunun hem boyut, hem konum, hem renk, hem de saydamlık değerleri belirli bir sırada gerçekleştirilmektedir.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.     $("button").click(function() {
8.         var div = $("div");
9.         div.animate({left: '200'}, "fast");
10.        div.animate({height: '300px', opacity: '0.4'},
11.           "slow");
12.        div.animate({width: '300px', opacity: '0.8'},
13.           "slow");
14.        div.animate({height: '100px', opacity: '0.4'},
15.           "slow");
16.        div.animate({width: '100px', opacity: '0.8'},
17.           "slow");
18.        div.animate({left: '10'}, "fast");
19.    });
20. });
21. </script>
22. </head>
23. <body>
24. <button>Animasyona başla!</button>
25. <p>Animasyonların belirli bir sırada
   gerçekleştirilemesi.</p>
26. <div
   style="background:#0000FF;height:100px;width:100px;posi
   tion:absolute;"></div>
27. </body>
28. </html>
```

Şekil 12.28. Animasyonların Belirli Bir Sırada Gerçekleştirilmesi.



Bireysel Etkinlik

- Proje klasörünüzün içinde "animasyon.html" adlı bir dosya oluşturun.
- Sayfada oluşturduğunuz bir HTML elemanın konumunu bir buton yardımıyla değiştirmeye çalışın. Butona her tıkladığınızda ilgili eleman ekranın 25 piksellik bir hareket gerçekleştirsin.

Durdurma (Stop)



jQuery'de animasyonlar normal sürelerinden önce durdurulabilirler.



stop() fonksiyonu iki farklı parametre alabilir.

jQuery'deki stop() fonksiyonu uygulama içerisindeki animasyonların henüz bitmeden önce durdurulması için kullanılır. stop() fonksiyonunun sözdizimi Şekil 12.29.'daki gibidir.

```
1. $(seçici).stop(tümünüDurdur, sonaGit);
```

Şekil 12.29. Stop() Fonksiyonu Sözdizimi.

jQuery'de stop() fonksiyonu iki farklı şekilde kullanılmaktadır. Bu kullanım şekilleri aşağıda listelenmiştir:

- **Parametresiz kullanım:** stop() fonksiyonu bu kullanımda herhangi bir parametre almaz. O an hangi animasyon gerçekleştiriliyorsa onu durdurur. Bir sonraki animasyonun başlamasını engellemez. Bir sonraki animasyonun durdurulması için ikinci bir tetiklemeye ihtiyaç duyulur.
- **Parametreli kullanım:** stop() fonksiyonu Şekil 12.29.'da gösterildiği gibi iki farklı parametre alabilir, bu parametrelerden ilki tüm animasyonların durdurulması ile ilgili parametredir. Tetikleme gerçekleştiği anda animasyon durdurulur ve eğer bir sonraki animasyon varsa bile o animasyona geçiş yapmaz. İkinci parametre ise animasyonu durdurur ve o animasyonun son karesine gider. Parametreleri aktif hâle getirmek için "true" (doğru) değerine fonksiyon içinde yer verilir.

stop() fonksiyonunun parametreli ve parametresiz kullanımı Şekil 12.30.'daki örnek uygulamada yer almaktadır.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <script src="jquery.js"></script>
5. <script>
6. $(document).ready(function() {
7.   $("#start").click(function() {
8.     $("div").animate({left: '100px'}, 5000);
9.     $("div").animate({fontSize: '2em'}, 5000);
10.  });
11.  $("#stop").click(function() {
12.    $("div").stop();
13.  });
14.  });
15.  $("#stop2").click(function() {
```

```
17.      $("div").stop(true);
18.  });
19.
20.  $("#stop3").click(function() {
21.    $("div").stop(true, true);
22.  });
23. });
24. </script>
25. </head>
26. <body>
27. <button id="start">Animasyona başla!</button>
28. <button id="stop">Animasyonu durdur!</button>
29. <button id="stop2">Tüm animasyonları durdur!</button>
30. <button id="stop3">Durdur ve sona git</button>
31. <p>stop() animasyonunun parametreli ve parametresiz
   kullanımı.</p>
32. <div
   style="background:#00FF00; height:100px; width:200px; posi
   tion:absolute;">Dur fonksiyonu</div>
33. </body>
34. </html>
```

Şekil 12.30 Stop() Fonksiyonu Örnek Uygulama.



:Özet

- jQuery, hızlı, küçük ve zengin özelliklere sahip bir JavaScript kütüphanesidir.
- jQuery, platform bağımsız olarak tanımlayabileceğimiz bir yapıda tasarlanmıştır. Bu yapısı birçok farklı İnternet tarayıcısında sorunsuz olarak çalışabilmesini sağlar.
- HTML belgesinde gezinme, olay işleme, animasyon ve diğer Ajax uygulamalarını çok daha basit hâle getirir.
- jQuery, açık kaynak kodlu bir projedir.
- JavaScript kütüphanelerinin genel çalışma prensipleri jQuery için de geçerlidir.
- jQuery kütüphane dosyasına iki farklı şekilde bağlantı sağlanabilir; dosya yerel bir adreste bulunabilir, İnternet üzerinde yer alan depolar üzerinden kullanılabilir.
- jQuery, kod geliştiriciler ve Web tasarımcılar için farklı kütüphane sürümleri sunar; bu sürümler arasındaki fark kodlar üzerinde değişiklik yapma kolaylığı sağlama ile ilgilidir.
- jQuery sözdizimi şablon temel olarak iki bölümden oluşur; seçici ve eylem. Seçici kısmı işlem yapılacak HTML elemanın seçilmesini, eylem kısmı ise gerçekleştirilecek fonksiyonu belirlememizi sağlar.
- jQuery Web sayfasının tamamı yüklenmeden çalışmaya başlaması durumunda hataya neden olabilir. Bu hatayı engellemek için "doküman hazır olayı" adlı özel bir fonksiyon yer almaktadır.
- jQuery'de bir fonksiyonun çalışması için farklı tetikleme seçenekleri bulunmaktadır. Bu seçenekler; fare olayları, klavye olayları, form olayları ve doküman olaylarıdır.
- jQuery içerisinde kullanıcı etkileşimiğini artıracak ve kullanıcı deneyimini daha eğlenceli hale getirebilecek efektler yer almaktadır. jQuery efektleri teker teker ya da bir arada kullanılabildiği gibi, birden fazla efekt belirli bir sırayla da kullanılabilir.
- Gizle / Göster efekti: jQuery ile HTML elemanlarını gizle (hide ve göster (show) metotları ile gizleyebilir ya da görünür hale getirebilirsiniz.
- Soldurma efekti: jQuery ile HTML elemanlarını soldurma (fade) metodunu kullanarak soldurma efekti ile gizleyebilir ya da belirginleştirme efekti ile görünür hale getirebilirsiniz.
- Kayma efekti: jQuery ile HTML elemanlarını kayma (slide) metodunu kullanarak kaydırabilirsiniz.
- Hareketlendirme efekti: jQuery ile HTML elemanlarına kendi belirlediğiniz şekilde (önceden tanımlanmış efektlerden bağımsız olarak) farklı hareket ve efektler vererek animasyonlar oluşturabilirsiniz. Bu animasyonlar HTML elemanlarının sayfa üzerindeki konumlarını değiştirerek, renk ve boyutları üzerinde değişiklikler yaparak gerçekleştirilmektedir.
- Hareketlendirme efekti uygulanmış çalışmalarda süreç tamamlanmadan animasyon durdurulmak isteniyorsa dur (stop) metodu kullanılabilir.

DEĞERLENDİRME SORULARI

1. HTML standartlarının ilk kullanılmaya başlandığı yıl aşağıdakilerden hangisidir?
 - a) 1991
 - b) 1993
 - c) 1998
 - d) 2003
 - e) 2005
2. Aşağıdakilerden hangisi HTML'in tek başına (herhangi bir ek bileşen olmadan) sahip olduğu özelliklerden biri değildir?
 - a) Farklı Internet tarayıcıları üzerinde çalışabilir.
 - b) Metin tabanlı bir etiketleme dili kullanılır.
 - c) Sayfa içerisindeki veri güncellemlerinde sayfanın yenilenmesine (refresh) gerek yoktur.
 - d) HTML özel olarak derlenmesine gerek kalmadan Internet tarayıcılarında yorumlanabilir.
 - e) HTML kaynak kodları herhangi bir metin editöründe görüntülenebilir.
3. jQuery kütüphane dosyalarının sıkıştırılmış sürümlerinin (min) sağladığı avantaj aşağıdakilerden hangisidir?
 - a) Boyut olarak daha küçük oldukları için Web sitesi optimizasyonu sağlar.
 - b) Dosyaların içeriğini herkes göremez, koruma sağlar.
 - c) Güncelleme işlemleri daha kolay gerçekleştirilir.
 - d) Farklı sunuculara aktarılması kolaydır.
 - e) Daha çok fonksiyonu içinde barındırabilir.
4. İçerik Dağıtım Ağı (Content Delivery Network – CDN) ne işe yarar?
 - a) Aynı içeriğin farklı Web sitelerinde yayınlanabilmesini sağlar.
 - b) Web sitelerindeki görsel ve işitsel materyalleri depolar.
 - c) Günlük gelişmelerin Internet üzerinden takip edilebileceği bir platform sunar.
 - d) İhtiyaç duyulan kaynak kodların Internet üzerinde yer alan depolardan kullanılabilmesini sağlar.
 - e) jQuery'de oluşturulmuş içeriklerin yedeklenmesini sağlar.

5. Doküman hazır olayı (The Document Ready Event) jQuery ile geliştirilen uygulamalar açısından neden önemlidir?
 - a) Dokümanın İnternet tarayıcıda daha hızlı yüklenmesini sağlar.
 - b) Yeni bir uygulama geliştirirken hazır olan jQuery kodlarını kullanmamızı sağlar.
 - c) İki farklı jQuery olayı arasında bağlantı kurar.
 - d) Web sitesinin jQuery ile uyumlu olup olmadığını kontrol eder.
 - e) Dokümanın yüklenmesi henüz tamamlanmadan jQuery kodlarının çalışmasına engel olarak olası bir hatayı engeller.
6. jQuery seçicileri Web sayfası içerisindeki hangi tür öğelerin seçiminde kullanılır?
 - a) JavaScript kodları içindeki fonksiyonların
 - b) HTML elemanlarının
 - c) Jpg, gif, png gibi görsel elemanların
 - d) Mp4, avi gibi akışkan medyanın
 - e) Mp3, wav gibi işitsel elemanların
7. Aşağıdakilerden hangisi jQuery fare olaylarından biri değildir?
 - a) Tıklama
 - b) Çift tıklama
 - c) Fare tekerleğinin döndürülmesi
 - d) Fare girişi
 - e) Fare çıkışı
8. Aşağıdakilerden hangisi jQuery efektleri içerisinde diğerlerine göre farklı bir işleve sahiptir?
 - a) Gizle / Göster
 - b) Soldurma
 - c) Kaydırma
 - d) Hareketlendirme
 - e) Durdurma
9. jQuery efektlerinde gerçekleştirilen işlemin aynı tetikleyici yardımıyla geri alınabilmesini sağlayan metot aşağıdakilerden hangisidir?
 - a) Show
 - b) Fade
 - c) Stop
 - d) Toggle
 - e) Slide

10. Açık kaynak kodlu bir proje olan jQuery hangi lisans şartları altında dağıtılmaktadır?
- a) MIT
 - b) GNU
 - c) Creative Commons
 - d) Telif Hakları
 - e) Apache

Cevap Anahtarı

1.b, 2.c, 3.a, 4.d, 5.e, 6.b, 7.c, 8.e, 9.d 10 -

YARARLANILAN KAYNAKLAR

Baltalı, S. (2011). jQuery. İstanbul: KODLAB

JQuery Öğrenme Merkezi. 19 Haziran 2019 tarihinde <https://learn.jquery.com/> adresinden erişildi.

JQuery Resmî Web Sitesi. 17 Haziran 2019 tarihinde <https://jquery.com/> adresinden erişildi.

Raggett, D., Lam, J., Alexander, I. ve Kmiec, M. (1999). History of HTML. Essex, England: Addison Wesley Longman Limited. 17 Haziran 2019 tarihinde <https://www.w3.org/People/Raggett/book4/ch02.html> adresinden erişildi.

Software Studio. MIT Courseware. 12 Temmuz 2019 tarihinde https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-170-software-studio-spring-2013/recitations/MIT6_170S13_rec6-jQuery.pdf adresinden erişildi.

W3Schools. jQuery Ders Notları. 3 Haziran 2019 tarihinde <https://www.w3schools.com/jquery> adresinden erişildi.

Web Programlama 302: jQuery. Turkcell Geleceği Yazanlar. 22 Haziran 2019 tarihinde <https://gelecegiyazanlar.turkcell.com.tr/konu/web-programlama/egitim/302> adresinden erişildi.

JQUERY MOBILE İLE TASARIM

İÇİNDEKİLER



- jQuery Mobile Nedir?
 - jQuery Mobile Paketi Neler İçerir?
 - jQuery Mobile Nasıl Çalışır?
- jQuery Mobile Kurulumu
- Mobil Web Sitesi Geliştirme Adımları
 - jQuery Mobile Sayfa Yapısı
 - jQuery Mobile Sayfa Bileşenleri
- jQuery Mobile Temaları

HEDEFLER

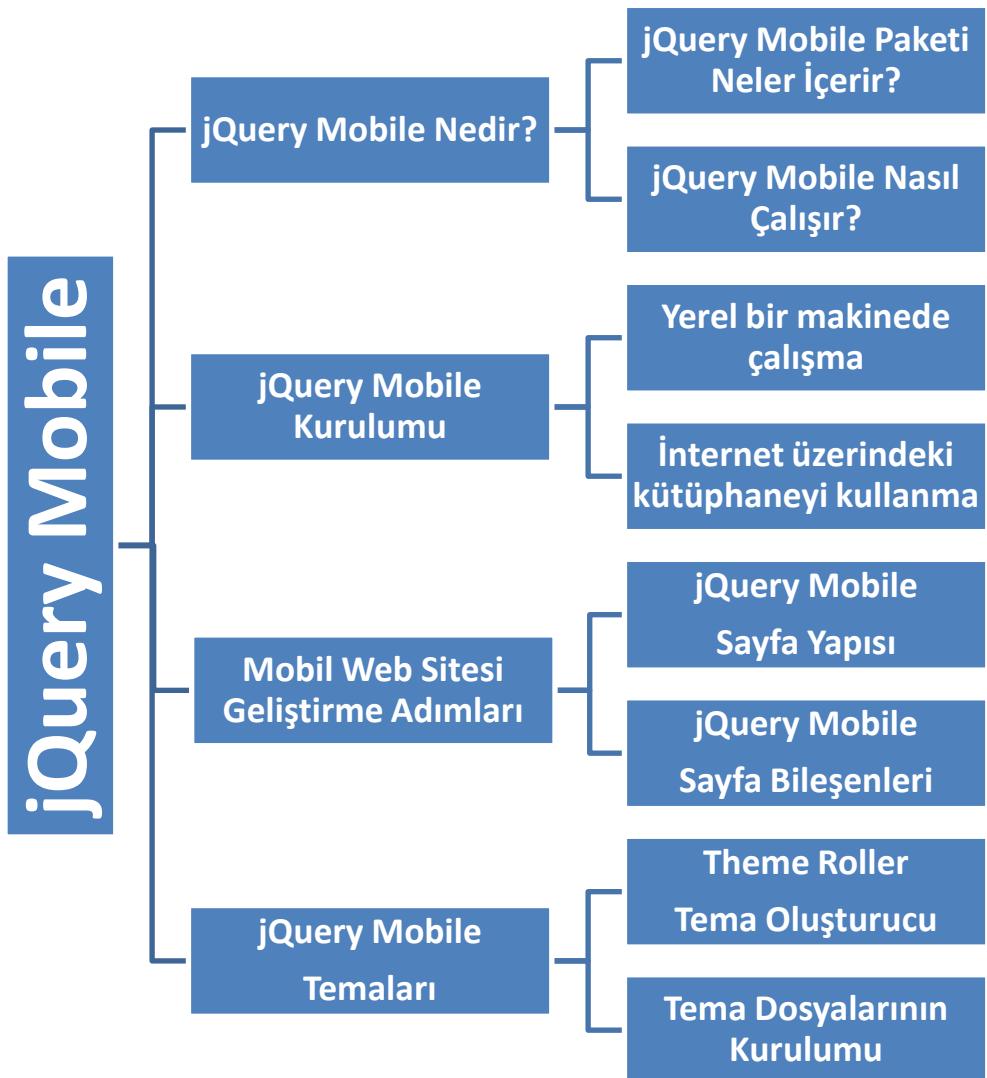
- Bu üniteyi çalıştıktan sonra;
 - jQuery Mobile kütüphanesinin kurulumunu yapabilecek,
 - jQuery Mobile kullanımı ile ilgili bilgi sahibi olacak,
 - jQuery Mobile tekli Web sayfası şablonu oluşturabilecek,
 - jQuery Mobile çoklu Web sayfası şablonu oluşturabilecek,
 - jQuery Mobile sayfa bileşenlerini oluşturabilecek ve farklı uygulamalarda kullanabilecek,
 - jQuery Mobile temaları oluşturabilecek ve bu temaları projelerinizde kullanabileceksiniz.



Atatürk Üniversitesi
Açıköretim Fakültesi

İNTernet
PROGRAMCILIĞI II
Öğr. Gör. Rafet
Orçun MADRAN

ÜNİTE
13



GİRİŞ

Internet, bilgiye erişim süreçleri üzerinde çok önemli değişikliklere yol açmıştır. Bilgi kaynaklarına erişim, ağ üzerinden gerçekleşmeye başladığı andan itibaren mekândan bağımsız olarak bilgiye erişim de mümkün hâle gelmiştir. Internet'e bağlı bir bilgisayardan çok farklı hizmetlere (e-devlet, e-finans, e-ticaret, vb.), kaynaklara erişilmesi ve bunun sağlamış olduğu avantajlar, kullanıcıların bu sürece mobil ortamda da devam etme taleplerini beraberinde getirmiştir. Bu talepleri mobil platformlarda karşılayabilmek için farklı yazılım çözümleri hayata geçirilmiştir.



Mobil cihazlarda bilişim hizmetlerine ulaşabilmek için mobil uygulamalar haricinde mobil internet tarayıcılar da kullanılabilir.

Mobil cihazlardan herhangi bir bilişim hizmetine erişebilmeniz için iki farklı yol bulunmaktadır. Bu yollardan ilki, ilgili hizmetin **mobil uygulamasını** uygulama marketlerinden (Apple App Store, Google Play vb.) cihazınıza indirmek ve cihaza bu uygulamayı kurarak kullanmaktadır. Diğer yol ise bilişim hizmetinin verildiği çevrim içi platforma mobil Internet tarayıcınız ile bağlanmaktadır. Her iki yolun da kendine göre avantaj ve dezavantajları bulunmaktadır. Internet üzerinden erişebileceğiniz kaynakların her zaman bir mobil uygulaması olmayabilir. Bu durumda web sitelerinin **Mobil Internet Tarayıcı** uyumlu olması önem kazanmaktadır. Özellikle ekran boyutu olarak küçük olan mobil cihazlardan tarayıcı üzerinde birtakım işlemler gerçekleştirmek eğer uygun tasarım bileşenleri kullanılmamışsa kimi zaman **eziyete** dönüşmektedir.

Mobil cihazlarıyla Web sitelerini ziyaret eden kullanıcıların iyi bir **kullanıcı deneyimi** yaşayabilmeleri için tercih edilen geliştirme ortamlarından biri jQuery Mobile'dır. **jQuery Mobile**, sadece mobil cihazlar için değil, dokunmatik ekrana sahip birçok farklı cihaz (kiosklar, araç navigasyon sistemleri vb.) için de uygulama geliştirme ortamı olarak kullanılabilir.

Bu ünite, mobil web programlama içerisinde ihtiyaç duyulabilecek navigasyon, animasyon ve sayfa içi etkileşimleri oluşturabileceğiniz kodları çok kolay bir şekilde yazabilmenizi sağlayacak altyapınızı sunmaktadır. Bu çalışmaları gerçekleştirebilmek için kullanacağımız **jQuery Mobile kütüphanesi** hakkında temel bilgiler, kurulum, kullanım şekilleri ve örnek uygulamalar da yine bu ünite içinde yer olacaktır.

JQUERY MOBILE NEDİR?

jQuery Mobile, **HTML5** temelli bir kullanıcı arayüz tasarım sistemidir. jQuery Mobile, tüm akıllı telefon, tablet ve masaüstü bilgisayarlar için uyumlu ve esnek web siteleri geliştirmenize olanak sağlar. jQuery'de olduğu gibi jQuery Mobile da slogan olarak "**write less, do more / az yaz, çok iş yap**" cümlesini kullanır. jQuery Mobile, özellikle sayfa içi etkileşim ve basit animasyonların gerçekleştirilebilmesini çok kolay hâle getirir. Bunlara ek olarak kullanıcı arayüzü oluşturacak özel olarak mobil cihazlar için yapılandırılmış HTML elemanları da içerir.

jQuery Mobile, açık kaynak kodlu bir projedir. Proje ile ilgili tüm bilgilere <https://jquerymobile.com/> adresinden ulaşılabilir. Projenin tüm kodları

MIT lisansı ile kullanıma sunulmuştur. MIT lisansının içeriği ve kullanım ile ilgili detaylı bilgi <http://ozgurlisanslar.org.tr/mit/> adresinde yer almaktadır.

jQuery Mobile Paketi Neler İçerir?

Programlama dillerinin standart yapılarına ek olarak kullanılabilecek özelliklerin içinde barındıran kod bloklarını *kütüphane* olarak tanımlayabileceğimizi bir önceki ünite (Ünite 12) belirtmiştik. jQuery Mobile, jQuery'nin aksine *sadece JavaScript kütüphanesinden oluşmaz*. HTML5 olarak ifade edilen, CSS ve JavaScript bileşenleri de jQuery Mobile paketinin içinde yer almaktadır. Bu paket arayüz tasarımda kullanılacak ek özelliklere göre farklı tema dosyaları da içermektedir.

jQuery Mobile Nasıl Çalışır?



jQuery Mobile sistem dosyaları yerel bir adreste yer alabileceği gibi, Internet üzerindeki bir depodan da kullanılabilir.

HTML5 çatılarının (framework) genel çalışma prensipleri jQuery Mobile için de geçerlidir. Aslında basit metin dosyalarından ibaret olan kütüphane dosyalarına (jQuery ve jQuery Mobile) ve CSS dosyasına HTML kodunun yazıldığı dosyadan bağlantı verilmesi yeterli olmaktadır. Bu dosyalara iki farklı şekilde bağlantı sağlanabilir:

- Dosyalar yerel bir adreste bulunabilir (çalışığınız bilgisayarın sabit diskinde ya da sunucunuzun herhangi bir dizininde).
- Dosyalar jQuery Mobile'in ve jQuery'nin Internet üzerinde yer alan depolarından kullanılabilir.

Yukarıda belirttiğimiz bağlantı şekilleri aynı zamanda jQuery Mobile'ın nasıl kurulacağı ile ilgili durumu da belirler. jQuery Mobile sistem paketinin sürekli *güncel* kalması isteniyorsa paketin Internet üzerindeki depolardan kullanılması uygun olacaktır. Ancak eğer geliştirilecek uygulama Internet'e bağlı olmayan bir platform için ise (örneğin bir kiosk sistemi ya da bir gömülü sistem) paketi oluşturan dosyaların *yerel* bir adreste bulunması gereklidir.

jQuery Mobile'ın nasıl çalıştığını gösteren kod bloku Şekil 13.1.'de gösterilmektedir. Şekil 13.1.'in 7., 8., ve 9. satırlarından yer alan adres tanımlaması, dosyaların yerel ya da Internet üzerinde bulunmasına bağlı olarak değişiklik gösterebilir.

```
1. <!doctype html>
2. <html>
3. <head>
4.   <meta charset="utf-8">
5.   <title>jQuery Mobile Test</title>
6.   <meta name="viewport" content="width=device-width,
initial-scale=1">
7.   <link rel="stylesheet" href="jquery.mobile.css" />
8.   <script src="jquery.js"></script>
9.   <script src="jquery.mobile.js"></script>
10.  </head>
11.  <body>
12.    <div data-role="page">
13.      <div data-role="header">
14.        <h1>jQuery Mobile Test</h1>
```

```
15.          </div>
16.          <div role="main" class="ui-content">
17.              Sayfa İçeriği
18.          </div>
19.          <div data-role="footer">
20.              <h4>Bu sayfa jQuery Mobile ile
21.                  oluşturulmuştur</h4>
22.          </div>
23.      </body>
24.  </html>
```

Şekil 13.1. Jquery'nin Çalışma Şekli, Örnek Kod Bloku.

JQUERY MOBİLE KURULUMU

jQuery Mobile ile ilgili kurulumda ihtiyaç duyacağımız tüm kaynak dosyalar <https://jquerymobile.com/download/> adresinde yer almaktadır. Bu sayfada jQuery Mobile dosyalarının hem sıkıştırılmış hem de sıkıştırılmamış sürümleri yer alır. *Sıkıştırılmamış* dosya biçimini daha çok *kod üzerinde geliştirme* çalışması yapmak isteyenler için hazırlanmıştır; okunması, üzerinde değişiklik yapılması daha kolaydır. *Sıkıştırılmış* dosya biçiminin avantajı ise *boyut olarak daha küçük* olmasıdır. Bu şekilde HTML sayfalarının mobil internet tarayıcıları tarafından yüklenmesi daha hızlı olur.

jQuery Mobile kullanımınız JavaScript kütüphanelerinin ve CSS tanımlamalarının hazır olan özelliklerini kullanmaktan ibare olacaksız ve çekirdek kod üzerinde herhangi bir değişiklik yapmayacaksanız, her zaman sıkıştırılmış jQuery Mobile sürümlerini tercih etmeniz gereklidir. Bu şekilde çalışmanız mobil web sitenizin genel *optimizasyonu* açısından önem taşımaktadır.

jQuery'de olduğu gibi jQuery Mobile dosya isimlendirmelerinde de belirli bir algoritma kullanılmaktadır. Bu sayede dosyanın adından ana sürüm bilgisi, alt sürüm bilgileri ve dosyanın biçimini hakkında bilgiler elde edilebilir. Aşağıdaki listede jQuery Mobile'ın son kararlı sürümü olan 1.4.5 için dosya isimlendirmeleri yer almaktadır:

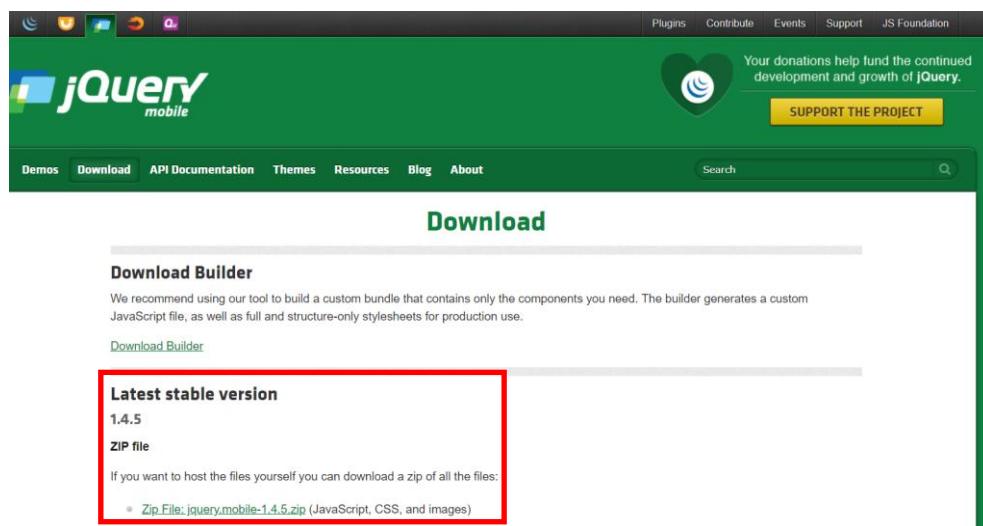
- jQuery Mobile 1.4.5 geliştirme sürümü, sıkıştırılmış: jquery.mobile-1.4.5.min.js
- jQuery Mobile 1.4.5 üretim sürümü, sıkıştırılmamış: jquery.mobile-1.4.5.js
- jQuery Mobile CSS tema dosyası sıkıştırılmış: jquery.mobile-1.4.5.min.css
- jQuery Mobile CSS tema dosyası sıkıştırılmamış: jquery.mobile-1.4.5.css

jQuery Mobile'ın temel özelliklerini kullanabilmemiz için *ikisi JavaScript* dosyası, *biri ise CSS* dosyası toplamda *3 adet dosyaya* ihtiyacımız bulunmaktadır. Bu dosyaların hepsini bir arada <http://jquerymobile.com/download/> adresinde yer alan "Latest stable version" (Son kararlı sürüm) başlığı altındaki "Zip File: jquery.mobile-1.4.5.zip" bağlantısına tıklayarak sabit diskinize indirebilirsiniz (Şekil 13.2.).



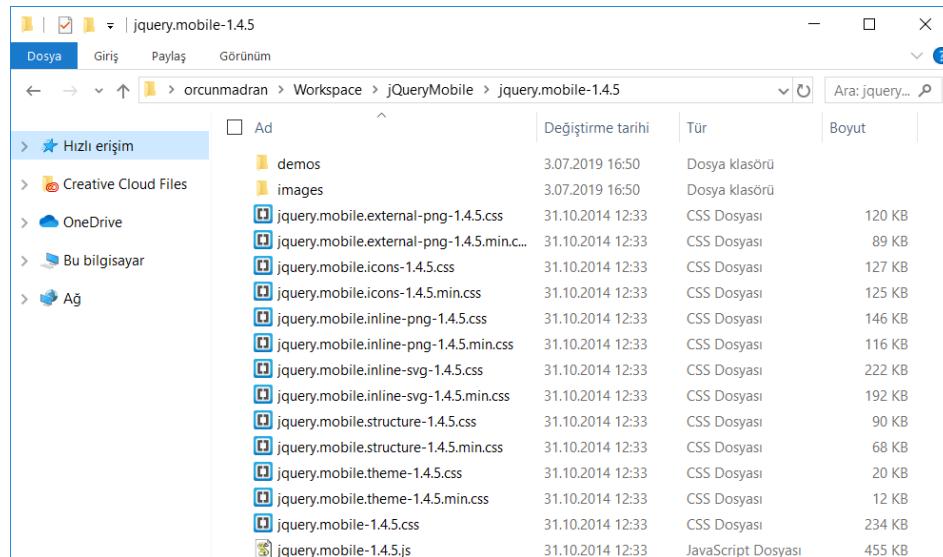
jQuery Mobile
kütüphane dosyaları iki
farklı biçimde
kullanılabilir:
sıkıştırılmış ya da
sıkıştırılmamış.

jQuery Mobile çekirdek paketi JavaScript kütüphane dosyalarından ve CSS tanımlama dosyasından oluşur.



Şekil 13.2. Jquery Mobile İndirme Sayfasında Bulunan Son Geçerli Sürüm Bağlantısı.

Sabit diskimize indirdiğimiz sıkıştırılmış dosyayı açtığımızda içinde yer alan dosyalar Şekil 13.3.'te görüldüğü gibidir. Bu paket içerisinde birçok farklı dosya ve klasör bulunmaktadır. Bu dosyalar ve klasörler jQuery Mobile ile gerçekleştirilebilecek farklı uygulamaların örnek dosyalarıdır. “*demos*” adlı klasörün içerisinde yer alan alt klasörlerden herhangi birine girerek “index.html” adlı dosyaya çift tıkladığınızda ilgili örnek internet tarayıcınızda görüntülenir.



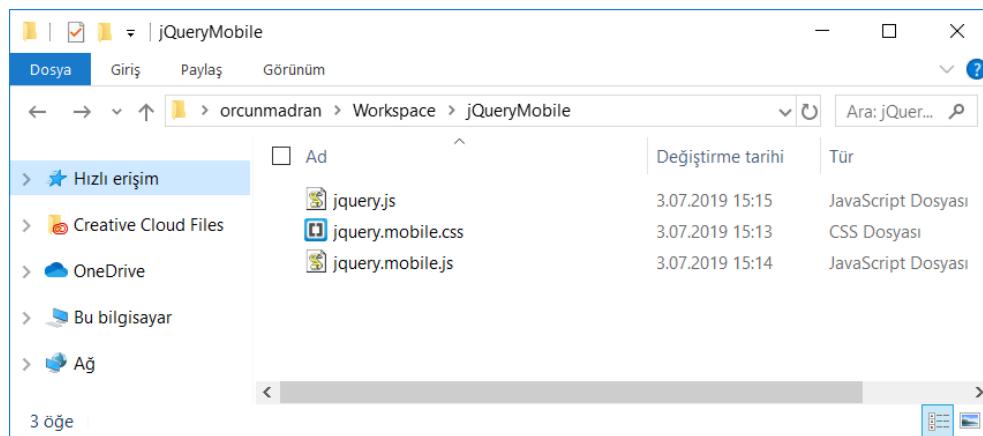
Şekil 13.3. Jquery Mobile Paket İçeriği Ve “Demo” Klasörü.

İhtiyacımız olan üç adet dosyayı elde edebilmemizin bir başka yolu da yine <https://jquerymobile.com/download/> adresinde bulunan ve Ünite 12'de de bahsettiğimiz jQuery'in CDN adı verilen *Internet depolarından* dosyaları indirmektir. CDN depolarında bulunan dosyaların adresleri ve sabit diskimizde “jQueryMobile” adlı proje klasörü içerisinde kaydederken kullanacağımız sadeleştirilmiş dosya adları Tablo 13.1.'de yer almaktadır.

Tablo 13.1. İndirilecek Dosyalar Ve Sadeleştirilmiş İsimleri.

	İndirilecek Dosya Bağlantısı	Sadeleştirilmiş Dosya Adı
jQuery Mobile CSS	http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.css	jquery.mobile.css
jQuery JS	http://code.jquery.com/jquery-1.11.1.min.js	jquery.js
jQuery Mobile JS	http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.js	jquery.mobile.js

Tablo 13.1.'de listelenen indirilecek dosyaları indirip adlarını sadeleştirerek **jQueryMobile** adlı proje klasöründe kaydettiğinizde bundan sonraki çalışmaları yapacağınız proje klasörünün görünümü Şekil 13.4.'teki gibi olmalıdır.



Şekil 13.4. "Jquerymobile" Proje Klasörü.

 jQuery Mobile projelerinde diğer Web projelerinde olduğu gibi tek bir ana klasör içinde çalışmak, dosya yapılandırmaları açısından faydalıdır.

Kurulum işleminin dosya yapılandırma aşamasını tamamladığımıza göre artık test için bir HTML dokümanı oluşturup **jQuery Mobile çekirdek dosyaları** ile bağlantısını kurabiliriz. Sabit diskimize oluşturduğumuz "jQueryMobile" adlı proje klasörümüzün içerisinde "test.html" adlı bir dosya oluşturalım ve içeriği de Şekil 13.5.'teki gibi olsun.

```

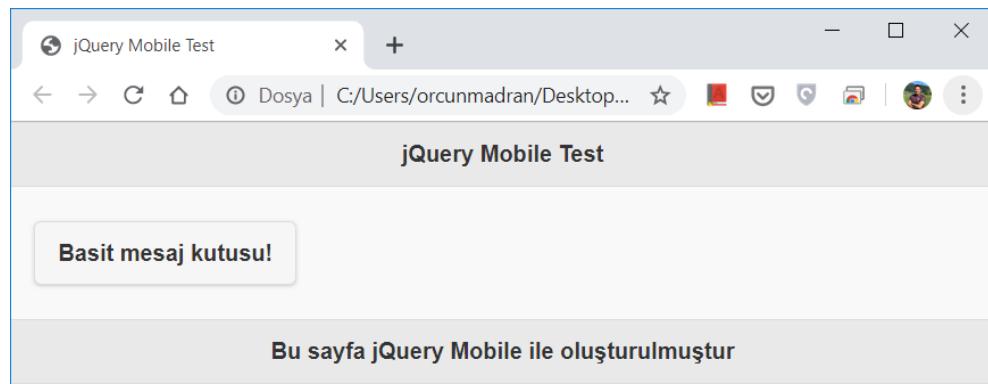
1. <!doctype html>
2. <html>
3. <head>
4.   <meta charset="utf-8">
5.   <title>jQuery Mobile Test</title>
6.   <meta name="viewport" content="width=device-width,
initial-scale=1">
7.   <link rel="stylesheet" href="jquery.mobile.css" />
8.   <script src="jquery.js"></script>
9.   <script src="jquery.mobile.js"></script>
10. </head>
11. <body>
12.   <div data-role="page">
13.     <div data-role="header">
14.       <h1>jQuery Mobile Test</h1>
15.     </div>
16.     <div role="main" class="ui-content">
17.       <a href="#popupBasic" data-
rel="popup" class="ui-btn ui-corner-all ui-shadow ui-

```

```
    btn-inline" data-transition="pop">Basit mesaj  
kutusu!</a>  
18.           <div data-role="popup"  
id="popupBasic">  
19.             <p>::: jQuery Mobile Basit Mesaj  
Kutusu :::</p>  
20.           </div>  
21.         </div>  
22.           <div data-role="footer">  
23.             <h4>Bu sayfa jQuery Mobile ile  
oluşturulmuştur</h4>  
24.           </div>  
25.         </div>  
26.   </body>  
27. </html>
```

Şekil 13.5. “Test.Html” İçinde Yer Alan Kod Bloğu.

Eğer kurulum işlemini eksiksiz gerçekleştirdiysek “test.html” dosyasına çift tıklayarak İnternet tarayıcısında açtığımızda Şekil 13.6.’daki gibi görünmelidir. jQuery Mobile’ın tüm fonksiyonlarının çalışıp çalışmadığını anlayabilmek için “Basit mesaj kutusu!” butonuna tıklayabilirsiniz. Butona tıkladığınızda bir mesaj kutusu çıkıyorsa tüm fonksiyonlar eksiksiz çalışıyor demektir.



Şekil 13.6. “Test.Html” Dosyasının İnternet Tarayıcısındaki Görüntüsü.

Bu örneğimizi jQuery Mobile çekirdek dosyalarını İnternet üzerindeki jQuery CDN deposundan bağlantı sağlayarak da test edebiliriz. Bu sayede ilgili dosyaların her zaman en güncel hâlini kullanmış oluruz. Şekil 13.6.’da yer alan test kodumuzun 7., 8., ve 9. satırlarını aşağıdaki gibi değiştirelim. Bu değişikliği yaptıktan sonra test.html dosyasını tarayıcıda açalım. *Internet bağlantımızın olması* gerektiğini de unutmayalım.

```
7. <link rel="stylesheet"  
href="http://code.jquery.com/mobile/1.4.5/jquery.mobile-  
1.4.5.min.css" />  
8. <script src="http://code.jquery.com/jquery-  
1.11.1.min.js"></script>  
9. <script  
src="http://code.jquery.com/mobile/1.4.5/jquery.mobile-  
1.4.5.min.js"></script>
```


jQuery Mobile projeleri
internet tarayıcılarında
ekstra bir bileşene
ihtiyaç duymadan
yorumlanabilir ve tüm
fonksiyonları ile
çalıştırılabilir.


İçerik Dağıtım Ağı (CDN)
sistemi, jQuery Mobile
kütüphanelerini sabit
diskinize kaydetmeden,
internet üzerinden
kullanmanızı sağlar.

jQuery Mobile ile ilgili kurulum aşamalarını tamamladık. Artık jQuery Mobile ile ilgili daha detaylı çalışmaları yapabilmek için gerekli altyapıya sahibiz.



Bireysel Etkinlik

- Sabit diskinizde "jQueryMobile" adlı bir proje klasörü oluşturun.
- Proje klasörünüzün içine jQuery Mobile çekirdek doslarını kaydedin.
- Şekil 13.5.'te yer alan kodları test.html dosyasına yerleştirerek Internet tarayıcıda görüntüleyin.

MOBİL WEB SİTESİ GELİŞTİRME ADIMLARI

jQuery Mobile ile bir uygulama geliştirmek temelde iki adımdan oluşmaktadır. Bu adımlar;

- jQuery Mobile sayfa yapısının oluşturulması,
- jQuery Mobile sayfa bileşenlerinin kullanılması.

jQuery Mobile Sayfa Yapısı

Bir jQuery Mobile sitesi, çatının (framework) tüm özelliklerinden tam olarak yararlanabilmek için bir *HTML5 doküman tipi* tanımlaması ile başlamalıdır.

Sayfanın <head> etiketleri içerisinde yer alan başlık bölümünde jQuery, jQuery Mobile ve CSS mobil temasına *referans verilmesi* zorunludur. Bu başlangıç gereksinimlerini kurulum aşamasında tamamlamıştık. jQuery Mobile'ın çalışma şekli üzerinde durduğumuz bölümde ise temelde kullanacağımız sayfa yapısını Şekil 13.1'de gözlemlemiştik. Şimdi ise Şekil 13.1'deki örnek kod yapısını detaylı bir şekilde inceleyelim.

Web sayfasının başlık bölümü

Web sayfasının başlık bölümünü <html> etiketinden sonraki ilk etiket olan <head> etiketi oluşturur. Sayfanın ilk yüklenmeye başladığı bölüm başlık bölümündür. Bu yüzden ihtiyaç duyulan tüm JavaScript kütüphanelerine ve CSS tanımlamalarına verilen referanslar (bir başka deyişle bağlantılar) başlık bölümünde yer alır. *Üst veri* (metadata) adı verilen ve Internet tarayıcısına beyan edilmek istenen diğer tanımlamalar da başlık bölümünde yer almaktadır. Sayfanın Internet tarayıcısında görüntülenen sayfa başlığı ve dil kodlaması hemen her sayfada bulunan üst veri alanları arasında yer alır.

jQuery Mobile özelinde kullanılan üst veri tanımlamalarından biri ise "*Viewport*" üst veri etiketidir. Bu etiket yardımıyla mobil Internet tarayıcıları masaüstü bir siteyi değil, mobil cihazlar için geliştirilmiş bir sayfayı görüntülediğini anlar ve cihazın ekranının piksel genişliğine göre içeriği *orantılı olarak* gösterir.



HTML dokümanlarında başlık bölümü sayfanın ilk yüklenen bölümündür, bu yüzden kütüphane dosyalarına bağlantılar bu bölümde yer verilir.

Standart bir jQuery Mobile web sitesinin sayfa yapısının başlık bölümü Şekil 13.7.'de gösterildiği gibi olmalıdır.

```
1. <head>
2.   <meta charset="utf-8">
3.   <title>jQuery Mobile Test</title>
4.   <meta name="viewport" content="width=device-width,
   initial-scale=1">
5.   <link rel="stylesheet" href="jquery.mobile.css" />
6.   <script src="jquery.js"></script>
7.   <script src="jquery.mobile.js"></script>
8. </head>
```

Şekil 13.7. Jquery Mobile Sayfası Yapısı Başlık Bölümü.

Web sayfasının gövde bölümü

Tüm web sayfalarında olduğu gibi mobil web sayfalarında da başlık bölümünden sonra gövde bölümü <body> gelir. Gövde bölümü web sayfasını ziyaret eden kullanıcının İnternet tarayıcısında göreceği tüm HTML elemanlarının yer aldığı bölümdür. Bu bölümde sayfa istenildiği gibi parçalara ayrılabilir ve farklı özelliklere göre (üst bilgi, alt bilgi vb.) tanımlamalar yapılabilir. Bu tanımlamalar <div> etiketi yardımıyla yapılır. Şekil 13.8.'de *üst bilgi*, *icerik* ve *alt bilgi* alanları tanımlanmış örnek bir sayfa yapısı yer almaktadır.

```
1. <body>
2.   <div data-role="page">
3.     <div data-role="header">...</div>
4.     <div role="main" class="ui-content">...</div>
5.     <div data-role="footer">...</div>
6.   </div>
7. </body>
```

Şekil 13.8. Örnek Sayfa Yapısı.

jQuery Mobile tek sayfa şablonu

Web sayfasının başlık ve gövde bölümlerini detaylı olarak incelediğimize göre artık bu bölümleri *tek bir sayfa* içerisinde birleştirebiliriz. Şekil 13.7. ve 13.8.'deki kodları tek bir HTML dokümanında birleştirdiğimiz zaman tarayıcıda elde ettiğimiz görüntü Şekil 13.9.'dakine benzer nitelikte olmalıdır.



Şekil 13.9. Örnek Web Sayfasının İnternet Tarayıcısındaki Görüntüsü.

Artık elimizde standart bir web sayfası tasarımda kullanabileceğimiz mobil uyumlu bir sayfa şablonu var. Web sitemizi oluştururken bu ya da benzeri şablon yapılarından kopyalar çıkartarak çalışmamızıza devam edebiliriz.



Web projelerinde, iyi kurgulanarak oluşturulmuş bir şablon dosyası tüm proje için iyi bir başlangıç noktası ve altyapı sağlayacaktır.



Bireysel Etkinlik

- Proje klasörünüz olan "jQueryMobile" içerisinde tekSayfa.html adlı bir dosya oluşturun.
- Bu dosya içerisinde Şekil 13.7. ve 13.8.'deki kod yapılarını kullanarak bir şablon yapısı oluşturacak kodları yazın.
- Gövde bölümüne istediğiniz şekilde içerikler girerek Şekil 13.9.'daki görünümü benzer bir görünümü İnternet tarayıcınızda elde etmeye çalışın.

jQuery Mobile çoklu sayfa şablonu

Günümüz Web sitelerinde çok kullanılan sayfa yapılandırmalarından biri, tek bir HTML sayfası içerisinde birden çok sayfanın yer almıştır. Aynı HTML dosyası içerisinde yukarı ve aşağı hareketler ile Web sitesi içerisinde dolaşmak mümkündür. Buna benzer bir çalışma mantığı jQuery Mobile içerisinde de uygulanabilir. **Tek bir HTML dokümanı** içerisine yerleştirilen **onlarca sayfa** Web sitesinin tamamını oluşturabilir.

Bu çalışma biçiminin bazı avantajları ve dezavantajları vardır. Tek bir sayfa içinde çalışıyor olmak daha derli toplu bir çalışma ortamı sunabilir, birçok HTML dokümanı ile ayrı ayrı uğraşmanız gereklidir, ancak unutmayın ki **tüm site aynı anda yüklenmektedir**. Hiç girilmeyecek sayfaların bile İnternet tarayıcısı tarafından yüklenmesi gerekmektedir. O açıdan Web sitesi tasarımda çoklu sayfa şablonu kullanımı Web sayfalarının adet olarak az olduğu durumlarda (ör: 4-7 arası) tercih edilmelidir.

Şekil 13.10.'da jQuery Mobile'da kullanılan çoklu sayfa şablonu örneği yer almaktadır. Şablonun başlık (`<head>`) bölümü tek sayfa şablonundan farklı değildir. Değişen bölümler sadece gövde içinde yer alan bölümlerdir.



Tek bir HTML dosyası içinde birden çok Web sayfasının barındırılması avantajlar sağlamağa ancak dosya boyutunu da artırmaktadır.

```

1. <body>
2.     <!-- Görüntülenecek ilk sayfa -->
3.     <div data-role="page" id="birinci">
4.         <div data-role="header">
5.             <h1>İlk Sayfa</h1>
6.         </div>
7.         <div role="main" class="ui-content">
8.             <p>Bu görüntülenen ilk sayfa</p>
9.             <p>İkinci sayfayı görüntülemek için lütfen
    <a href="#ikinci">tıklayın</a>!</p>
10.            </div>
11.            <div data-role="footer">
12.                <h4>jQuery Mobile</h4>
13.            </div>
14.        </div>
15.        <!-- İkinci sayfa -->
16.        <div data-role="page" id="ikinci">
17.            <div data-role="header">
18.                <h1>İkinci Sayfa</h1>
19.            </div>
20.            <div role="main" class="ui-content">
21.                <p>Şimdi ikinci sayfayı
    görüntüleyorsunuz.</p>
22.                <p>İlk sayfaya dönmek için <a
    href="#birinci">tıklayın</a>!</p>
23.            </div>
24.            <div data-role="footer">
25.                <h4>jQuery Mobile</h4>
26.            </div>
27.        </div>
28.    </body>

```

Şekil 13.10. Çoklu Sayfa Şablonu Örneği.

Şekil 13.10.'daki örnek kodları incelediğimizde gövde içerisinde yer alan iki farklı sayfa yapısı olduğunu görüyoruz. 1. sayfa yapısı satır 3-14 arasında, 2. sayfa yapısı ise satır 16-27 arasında yer almaktadır. İki sayfa yapısı arasındaki en önemli fark **3. ve 16. satırlardaki "id" tanımlamasıdır**. Bu tanımlama her bir sayfanın benzeriz kimliğini göstermelidir. Örneğimizde ilk sayfa için "birinci", ikinci sayfa için "ikinci" anahtar kelimeleri kimlik tanımlaması olarak kullanılmıştır. jQuery Mobile her zaman ilk sayfa tanımlamasını öncelikli olarak dikkate almakta ve tarayıcıda o görüntülemektedir. Sonraki sayfa yapıları bir anlamda gizlenmektedir. Tek bir HTML dosyası içerisinde istediğiniz kadar sayfa yapısı oluşturulabilir. Dikkat etmeniz gereken sayfa yapılarına **farklı "id" tanımlamaları** yapmanız gerektiğiidir.

Sayfalar arası geçiş işlemleri ise HTML içerisinde bağlantı vermek için standart olarak kullanılan `<a>` etiketi ile sağlanır. Ancak bu sefer `<a>` etiketi içerisinde verilecek bağlantı **sayfa içi bağlantı** olacaktır ve "#sayfaID" şeklinde bir tanımlamaya sahip olmalıdır. Bu tanımlama Şekil 13.10.'daki örnek kodun 9. ve 22. satırlarında yer almaktadır.



Tek sayfadan oluşan Web sitesi tasarımlarında sayfa arası geçişler, sayfa içi bağlantı verme metodu ile gerçekleşir.



Bireysel Etkinlik

- Proje klasörünüz olan "jQueryMobile" içerisinde cokluSayfa.html adlı bir dosya oluşturun.
- Bu dosya içerisinde Şekil 13.10.'daki kod yapılarını kullanarak çoklu sayfa şablonu oluşturun.
- Sayfalar arasında <a> etiketi ve "id" (#) tanımlaması ile geçiş yapın.



jQuery Mobile sayfa geçiş efektleri, kullanıcıların Web sitesi gezintilerinde mobil uygulama deneyimi yaşamalarını sağlayabilir.

jQuery Mobile Sayfa Geçiş Efektleri

jQuery Mobile içerisinde bir sayfadan diğer bir sayfaya geçişte kullanılabilecek farklılıklere sahip geçiş efektleri (transitions) bulunmaktadır. Bu geçiş efektleri kullanıcı deneyimi açısından önemli bir avantaj sağlar; kullanıcılar sanki cihaza yüklü bir mobil uygulama kullanmış hissine kapılırlar. Sayfa geçişleri temelde iki şekilde yapılır:

- Yeni sayfanın bir iletişim penceresi içinde açılması (dialog),
- Yeni sayfanın mevcut sayfanın yerinde açılması (page).

Sayfa geçişlerinde kullanılabilen efekt çeşitleri ise;

- Soldurma (fade),
- Açılan pencere (pop),
- Ters-yüz (flip),
- Dönerek (turn),
- Akarak (flow),
- Kayarak ve soldurarak (slidefade),
- Kayarak (slide),
- Yukarı kayarak (slideup),
- Aşağı kayarak (slidedown),
- Efektsiz geçiştir (none).

Sayfa geçişinde kullanılacak kod yapısı Şekil 13.11.'de yer almaktadır.

- ```
1. <a href="tekSayfa.html" data-rel="dialog" data-
 transition="flip" class="ui-btn ui-corner-all ui-shadow
 ui-btn-inline">İletişim kutusunda aç!
2. <a href="tekSayfa.html" data-transition="flip"
 class="ui-btn ui-corner-all ui-shadow ui-btn-
 inline">Sayfa olarak aç!
```

Şekil 13.11. Sayfa Geçişi Örnek Kodları.

Şekil 13.11'deki ilk bağlantı kodu, açılacak olan sayfanın *İletişim kutusu* içerisinde açılmasını, ikinci bağlantı kodu ise açılacak sayfanın *normal olarak*

açılmamasını sağlamaktadır. Kodları yakından incelediğimizde ilk bağlantı kodunda yer alan **“data-rel=“dialog”** tanımlamasının iletişim kutusunu aktif hâle getirdiği anlaşılmaktadır. Eğer bu tanımlama yapılmazsa (ikinci bağlantıkı olduğu gibi), geçiş efekti uygulanır ve sayfa normal olarak açılır.

Her iki bağlantı şeklinde de “**data-transition**” parametresi sayfa geçisi ile ilgili efekti tanımlamamızı sağlar. Yukarıdaki efekt listesinden herhangi bir efekt seçilerek (İngilizce anahtar kelimeler) istenilen şekilde sayfa geçisinin gerçekleşmesi sağlanabilir.



Sayfa geçisi ile ilgili farklı efektleri aynı örnek üzerinde denemek için “**data-transition**” parametresini değiştirmeniz yeterli olacaktır.



### Bireysel Etkinlik

- Proje klasörünüz olan "jQueryMobile" içerisinde tekSayfa.html dosyasını şablon olarak kullanarak iki farklı dosya oluşturun.
- Bu dosyalar arasındaki sayfa geçisini Şekil 13.11.'deki bağlantı kodları yardımıyla sağlayın.
- **data-transition** parametresini değiştirerek farklı geçiş efektlerini internet tarayıcınızda test edin.

## jQuery Mobile Sayfa Bileşenleri

jQuery Mobile sayfa içerisinde birçok farklı HTML elemanını ve bu elemanlar yardımıyla oluşturulmuş nesneleri kullanmamıza olanak sağlar. Bu nesneler özel olarak jQuery Mobile için yapılandırılmışlardır ve **bileşen** (widget) olarak ifade edilirler.

### Panel bileşeni



Panel bileşenleri açılır-kapanır yapıları ile sayfa içerisinde ek bir yer kaplamazlar.

jQuery Mobile panel bileşeni, hem site içi **navigasyonun** sağlanması amacıyla, hem de ilgili sayfada **ek bilgiler** vermek amacıyla kullanılabilir. Panel bileşeninin en önemli avantajı **açılır-kapanır** bir yapıda oluşudur. Bu sayede ekranda ayrıca bir **yer kaplamaz**, sadece ihtiyaç duyulduğu anda aktive edilebilir. Paneller üç farklı şekilde açılıp ve kapanırlar:

- Üstüne açılan (overlay),
- Sayfayı kaydırın (reveal),
- İterek açılan (push).

Şekil 13.12.'de sayfanın üstüne doğru açılan (overlay) panel örneği yer almaktadır.

```
1. <body>
2. <div data-role="page">
3. <div data-role="panel" id="detayliBilgi" data-
position="left" data-display="overlay">
4. <h3>Detaylı Bilgi</h3>
5. <p>jQuery ile ilgili detaylı bilgiye
burada yer verebilirisiniz.</p>
6. <a href="#" data-rel="close" class="ui-
btn ui-shadow ui-corner-all ui-icon-delete ui-btn-icon-
left ui-btn-inline">Paneli kapat!
7. </div>
```

```

8. <div data-role="header">
9. <h1>jQuery Mobile'a Hoşgeldiniz</h1>
10. </div>
11. <div role="main" class="ui-content">
12. <p>jQuery Mobile, HTML5 temelli bir
kullanıcı arayüz tasarım sistemidir. jQuery Mobile, tüm
akıllı telefon, tablet ve masaüstü bilgisayarlar için
uyumlu ve esnek web siteleri geliştirmenize olanak
sağlar.</p>
13. <p>jQuery'de olduğu gibi jQuery Mobile da
slogan olarak "write less, do more / az yaz, çok iş
yap" cümlesini kullanır. jQuery Mobile, özellikle sayfa
içi etkileşim ve basit animasyonların
gerçekleştirilebilmesini çok kolay hale getirir.</p>
14. <p><a href="#detayliBilgi" class="ui-btn
ui-shadow ui-corner-all ui-btn-inline ui-btn-
mini">Detaylı bilgi için tıklayın!</p>
15. </div>
16. <div data-role="footer">
17. <h4>jquerymobile.com</h4>
18. </div>
19. </div>
20. </body>

```

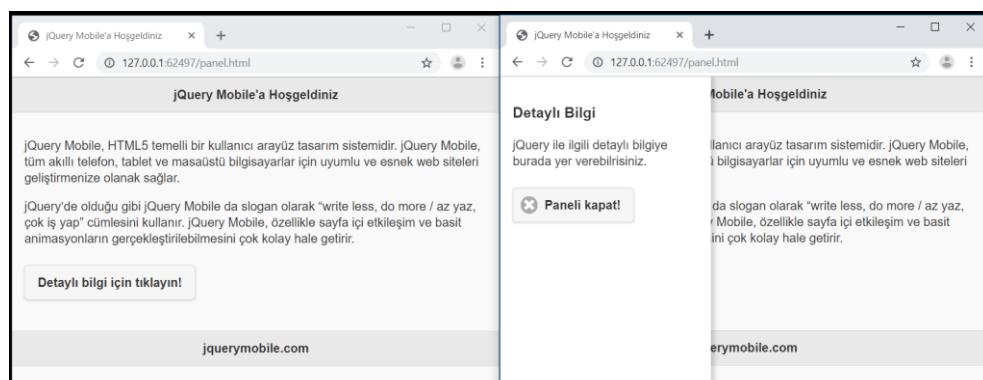
Şekil 13.12. Sayfanın Üstüne Açılan Panel Örneği.

Şekil 13.12.'deki kodları incelediğimizde panel ile ilgili tanımlamaların 3. satırda yer aldığığini görüyoruz. Bu satırda ***data-role="panel"*** ifadesi bu ögenin bir panel olduğunu, ***id="detayliBilgi"*** ifadesi panelin kimliğini, ***data-position="left"*** ifadesi panelin soldan açılacağını, ***data-display="overlay"*** ifadesi ise panelin açılma şeklini tanımlamaktadır. Buradaki parametreler üzerinde değişiklikler yapılarak örneğin panelin sağdan açılması (***data-position="right"***) ya da panelin sayfayı iterek açılması (***data-display="push"***) sağlanabilir.

Şekil 13.13.'te, Şekil 13.12.'de yer alan panel örneğinin İnternet tarayıcıda görüntülenmiş hâli bulunmaktadır. Soldaki ekran görüntüsünde panelin kapalı hâli, sağdaki ekran görüntüsünde ise panelin açık hâli yer almaktadır.



Bir sayfada birden fazla panel kullanılacaksa, kullanılacak 2. panelin kodları sayfanın alt tarafına yazılmalıdır.



Şekil 13.13. Panel Örneği Ekran Görüntüleri.

**Bireysel Etkinlik**

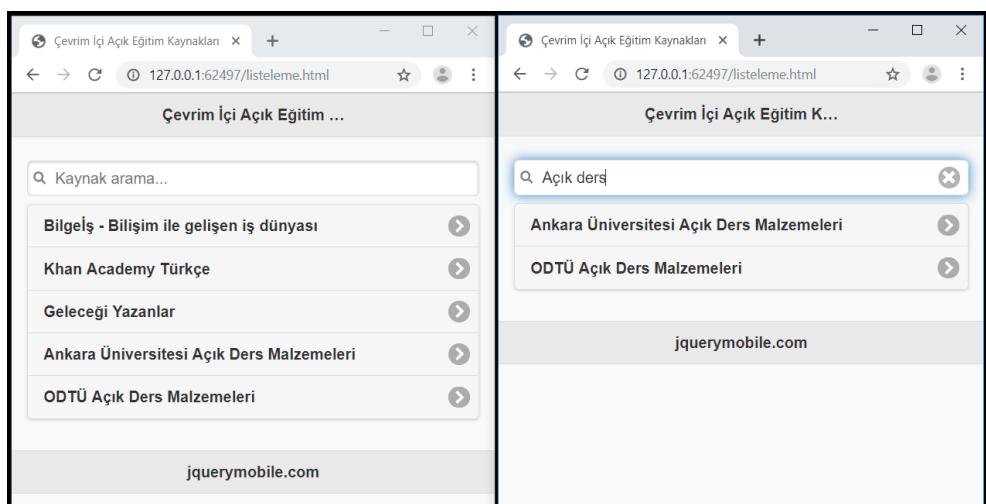
- Proje klasörünüz olan "jQueryMobile" içerisinde panel.html adlı bir dosya oluşturun.
- Şekil 13.11.'deki örnek kodlar yardımıyla sağdan ya da soldan istediğiniz tarzda açılan bir panel oluşturun.
- Bu panelin içeriğini farklı HTML elemanları ile doldurarak Internet tarayıcınızda test edin.

**Liste bileşeni**

Web sitelerinde *veri gösterimi* ile ilgili çalışmalarında listeleme özelliği sıkça kullanılan özellikler arasında yer almaktadır. Verilerin listelenmesi ve listelenen veriler arasında *filtreleme yoluyla arama* yapılması jQuery Mobile içerisinde kolaylıkla gerçekleştirilebilir. Standart HTML elemanlarından olan *madde imleri* bu çalışmanın ana hatlarını teşkil etmektedir. Madde imleri jQuery Mobile CSS temaları ile hem şekil değişikliğine uğramakta hem de jQuery kütüphanesinin\_filtreleme özelliği ile *liste içinde arama fonksiyonu* aktif hâle getirilebilmektedir.

Listeme ve filtrelere özelliğini bir arada kullanabileceğimiz bir örnek üzerinde çalışalım. Örnek çalışmanın Internet tarayıcıındaki görüntüsü Şekil 13.14.'teki gibi olsun. Soldaki ekran görüntüsünde listenin tamamı yer almaktadır. Sağdaki ekran görüntüsünde ise arama kutusuna girilen 2-3 harften sonra sadece o harfler ile eşleşen liste elemanları görüntülenmektedir. *Arama fonksiyonu* bir anlamda *filtreleme* olarak çalışmaktadır.

  
Listeleme çalışmalarında kullanabileceğimiz filtrelere özelliği basit bir arama fonksiyonu sağlar.



**Şekil 13.14.** Listeleme Ve Filtreleme Örnek Uygulaması Ekran Görüntüleri.

Şekil 13.14.'teki örnek uygulamanın kodları Şekil 13.15.'te yer almaktadır. Örnek kodları incelediğimizde satır 7-13 arasında `<ul>` etiketi ile madde imi (unordered list) tanımlaması yapıldığı görülmektedir. Listeye yeni bir eleman eklenmek istenildiğinde `<ul></ul>` etiketleri arasında yer alan bölüme `<li>` etiketi ile yeni bir liste ögesi eklenebilir.

```

1. <body>
2. <div data-role="page">
3. <div data-role="header">
4. <h1>Çevrim İçi Açık Eğitim Kaynakları</h1>
5. </div>
6. <div role="main" class="ui-content">
7. <ul data-role="listview" data-filter="true"
 data-filter-placeholder="Kaynak arama..." data-
 inset="true">
8.
 Bilgeiş - Bilişim ile gelişen iş
 dünyası
9.
 Khan Academy Türkçe
10.
 Geleceği Yazanlar
11.
 Ankara Üniversitesi Açık Ders
 Malzemeleri
12.
 ODTÜ Açık Ders Malzemeleri
13.
14. </div>
15. <div data-role="footer">
16. <h4>jquerymobile.com</h4>
17. </div>
18. </div>
19. </body>

```

**Şekil 13.15.** Listeleme Ve Filtreleme Örnek Kodları.

Şekil 13.15.'teki kodları daha detaylı bir şekilde incelediğimizdefiltreleme işleminin 7. satırda tanımlandığını anlıyoruz. Buradaki ***data-role="listview"*** ifadesi madde imlerinden oluşan yapının bir liste görünümüne kavuşturmasını sağlıyor. ***data-filter="true"*** ifadesi bir arama kutusunu listenin üst bölümüne ekliyor (eğer bu değer "false" yapılrsa arama kutusu kaybolacaktır). ***data-filter-placeholder="Kaynak arama..."*** ifadesi ise arama kutucuğundaki varsayılan mesajı tanımlıyor.



Listeleme çalışmalarında filtre özelliği kullanılmak istenmiyorsa "data-filter" parametresine "false" değeri girilir.



#### Bireysel Etkinlik

- Proje klasörünüz olan "jQueryMobile" içerisinde liste.html adlı bir dosya oluşturun.
- Şekil 13.15.'teki örnek kodlar yardımıyla basit bir sözlük oluşturmaya çalışın.
- Oluşturduğunuz sözlük içerisinde\_filtreleme yoluyla arama yapılabilisin ve ilgili sözcüğe tıklandığında kullanıcı detaylı bilgiye ulaşabilisin (örn: Türk Dil Kurumunun ilgili sayfalarına yönlendirme yapabilirsiniz).

## Tablo bileşeni

Web sitelerinde *veri gösterimi* ile ilgili çalışmalarda sıkılıkla kullanılan bir diğer bileşen ise tablo bileşenidir. Tablo bileşeni yardımıyla veriler birden çok sütunda yayınlanabilirler ve istenilen *sütunlar* aktif hâle getirilerek *görüntülenebilir* ya da devre dışı bırakılarak *görüntülenmeleri engellenebilir*. Bu sayede özellikle küçük ekranlarda daha iyi bir kullanıcı deneyimi sağlanabilir.



jQuery Mobile'da oluşturulacak tablolarda tüm sütunlar görüntülenebilir ya da bazı sütunlar gizlenebilir.

Tablo özelliğini gözlemleyebileceğimiz bir örnek üzerinde çalışalım. Örnek çalışmanın İnternet tarayıcısındaki görüntüsü Şekil 13.16.'daki gibi olsun. Soldaki ekran görüntüsünde tablodaki sütunların bazıları yer almaktadır. Sağdaki ekran görüntüsünde ise tablodaki tüm sütunlar seçilmiş olarak görüntülenmektedir.

| Kaynak Kısık Adı  | Web Adresi                       | Tahmini Ders Sayısı | Kaynak Kısık Adı  | Kaynak Uzun Adı                           | Web Adresi                       | Sayı |
|-------------------|----------------------------------|---------------------|-------------------|-------------------------------------------|----------------------------------|------|
| Bilgelş           | bilgeis.net                      | 100+                | Bilgelş           | Bilgelş - Bilişim ile gelişen iş dünyası  | bilgeis.net                      | 100+ |
| KA                | www.khanacademy.org.tr           | 200+                | KA                | Khan Akademii Türkçe                      | www.khanacademy.org.tr           | 200+ |
| Geleceği Yazarlar | gelecegiyazarlar.turkoell.com.tr | 50+                 | Geleceği Yazarlar | Turkcell Geleceği Yazarlar                | gelecegiyazarlar.turkoell.com.tr | 50+  |
| Ankadem           | acikders.ankara.edu.tr           | 500+                | Ankadem           | Ankara Üniversitesi Açık Ders Malzemeleri | acikders.ankara.edu.tr           | 500+ |
| ODTÜ ADM          | ocw.metu.edu.tr                  | 500+                |                   |                                           |                                  |      |
| jquerymobile.com  |                                  |                     |                   |                                           |                                  |      |

Şekil 13.16. Tablolama Örnek Uygulaması Ekran Görüntüleri.

Şekil 13.16.'daki örnek uygulamanın kodları Şekil 13.17.'de yer almaktadır. Örnek kodları incelediğimizde satır 8-15 arasında tablonun başlık bölümünün, satır 16-47 arasında ise tablonun veri bölümünün yer aldığı görülür.

```

1. <body>
2. <div data-role="page">
3. <div data-role="header">
4. <h1>Çevrim İçi Açık Eğitim Kaynakları</h1>
5. </div>
6. <div role="main" class="ui-content">
7. <table data-role="table" id="table-column-
 toggle" data-mode="columntoggle" class="ui-responsiv-
 e-table-stroke">
8. <thead>
9. <tr>
10. <th>Kaynak Kısık Adı</th>
11. <th data-priority="2">Kaynak Uzun
 Adı</th>
12. <th data-priority="1">Web
 Adresi</th>
13. <th data-priority="3">Tahmini Ders
 Sayısı</th>
14. </tr>
15. </thead>
16. <tbody>
17. <tr>
18. <td>Bilgelş</td>
19. <td>Bilgelş - Bilişim ile gelişen iş
 dünyası</td>

```

```

20. <td>bilgeis.net</td>
21. <td>100+</td>
22. </tr>
23. <tr>
24. <td>KA</td>
25. <td>Khan Akademi Türkçe</td>
26. <td>www.khanacademy.org.tr</td>
27. <td>200+</td>
28. </tr>
29. <tr>
30. <td>Geleceği Yazanlar</td>
31. <td>Turkcell Geleceği Yazanlar</td>
32.
33. <td>gelecegiyazanlar.turkcell.com.tr</td>
34. <td>50+</td>
35. </tr>
36. <td>Ankadem</td>
37. <td>Ankara Üniversitesi Açık Ders
 Malzemeleri</td>
38. <td>acikders.ankara.edu.tr</td>
39. <td>500+</td>
40. </tr>
41. <tr>
42. <td>ODTÜ ADM</td>
43. <td>Orta Doğu Teknik Üniversitesi
 Açık Ders Malzemeleri</td>
44. <td>ocw.metu.edu.tr</td>
45. <td>500+</td>
46. </tr>
47. </tbody>
48. </table>
49. </div>
50. <div data-role="footer">
51. <h4>jquerymobile.com</h4>
52. </div>
53. </div>
54. </body>

```

Şekil 13.17. Tablolama örnek kodları.

Şekil 13.17.'deki kodlar detaylı bir şekilde incelendiğinde tablo yapısı ile ilgili tanımlamaların 7. satırda yapıldığı görülmektedir. Bu satırdaki ***data-mode="columntoggle"*** ifadesi tablonun sütunlarının seçilebilir olmasını sağlayan tanımlamayı yapmaktadır.



Tablodaki sütunların seçilebilir olmasını istemiyorsak "data-mode" parametresini silerek bu özelliği devre dışı bırakabiliriz.



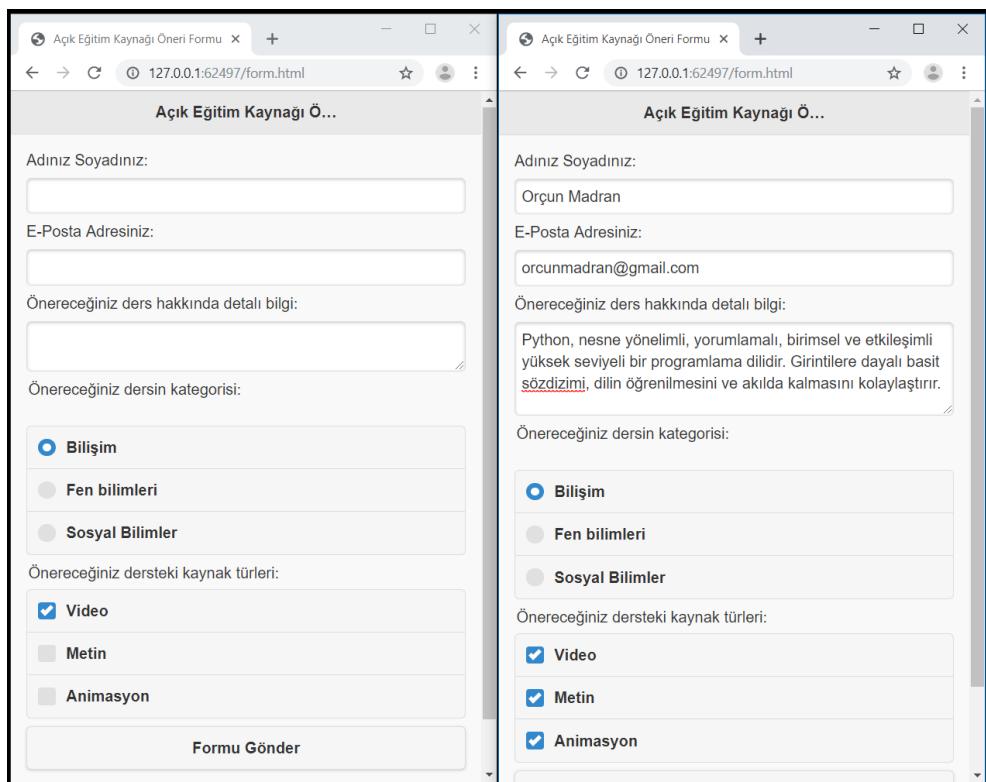
#### Bireysel Etkinlik

- Proje klasörünüz olan "jQueryMobile" içerisinde tablolama.html adlı bir dosya oluşturun.
- Şekil 13.17.'deki örnek kodlar yardımıyla 5 sütunu olan bir tablo oluşturun.
- Oluşturduğunuz tablo içerisindeki sütunların eklenip çıkarılabilir olmasını sağlayın ve İnternet tarayıcınızda bu fonksiyonu test edin.

## Form bileşenleri

jQuery Mobile içerisinde kullanıcının Web sayfalarında gerçekleştireceği *veri girişi* ile ilgili çok çeşitli form bileşenleri bulunmaktadır. Bu form bileşenleri standart *HTML form elemanlarının* (metin kutuları, onay kutuları, radyo düğmeleri vb.) jQuery Mobile'ın CSS tanımlamaları ile *özel olarak biçimlendirilmiş* hâlleridir. Bunlara ek olarak birtakım özel bileşenler de (örn: kaydırıcılar (slider), şalter görünümülü düğmeler (flip switch)) form bileşenleri altında kullanılabilmektedir.

Şekil 13.18.'deki ekran görüntüleri en çok kullanılan form bileşenlerinden oluşturulmuş bir form çalışmasına aittir. Formda basit metin kutuları ve detaylı metin alanı, radyo butonları, onay kutuları ve formun gönderilmesi için bir düğme (buton) yer almaktadır.



Şekil 13.18. Form Örnek Uygulama Ekran Görüntüleri.

Şekil 13.18.'deki örnek uygulamanın kodları şekil 13.19.'da yer almaktadır.

```

1. <body>
2. <div data-role="page">
3. <div data-role="header">
4. <h1>Açık Eğitim Kaynağı Öneri Formu</h1>
5. </div>
6. <div role="main" class="ui-content">
7. <label for="text-basic">Adınız Soyadınız:</label>
8. <input type="text" name="text-basic" id="text-basic" value="">
9. <label for="text-basic">E-Posta Adresiniz:</label>
10. <input type="text" name="text-basic" id="text-basic" value="">
```

```

11. <label for="textarea">Önereceğiniz ders
 hakkında detali bilgi:</label>
12. <textarea cols="40" rows="8"
 name="textarea" id="textarea"></textarea>
13. <fieldset data-role="controlgroup">
14. <legend>Önereceğiniz dersin
 kategorisi:</legend>
15. <input type="radio" name="radio-
 choice-1" id="radio-choice-1" value="choice-1"
 checked="checked">
16. <label for="radio-choice-
 1">Bilişim</label>
17. <input type="radio" name="radio-
 choice-1" id="radio-choice-2" value="choice-2">
18. <label for="radio-choice-2">Fen
 bilimleri</label>
19. <input type="radio" name="radio-
 choice-1" id="radio-choice-3" value="choice-3">
20. <label for="radio-choice-3">Sosyal
 Bilimler</label>
21. </fieldset>
22. <fieldset data-role="controlgroup">
23. <legend>Önereceğiniz dersteki kaynak
 türleri:</legend>
24. <input type="checkbox"
 name="checkbox-1a" id="checkbox-1a" checked="">
25. <label for="checkbox-
 1a">Video</label>
26. <input type="checkbox"
 name="checkbox-2a" id="checkbox-2a">
27. <label for="checkbox-
 2a">Metin</label>
28. <input type="checkbox"
 name="checkbox-3a" id="checkbox-3a">
29. <label for="checkbox-
 3a">Animasyon</label>
30. </fieldset>
31. <button class="ui-shadow ui-btn ui-
 corner-all">Formu Gönder</button>
32. </div>
33. <div data-role="footer">
34. <h4>jquerymobile.com</h4>
35. </div>
36. </div>
37. </body>

```



Tüm form elemanları ve bileşenleri veri gönderimi yapabilmek için form etiketi `<form>` içinde yer almalıdır.

## Formlar

**Şekil 13.19.** Form Örnek Kodları.

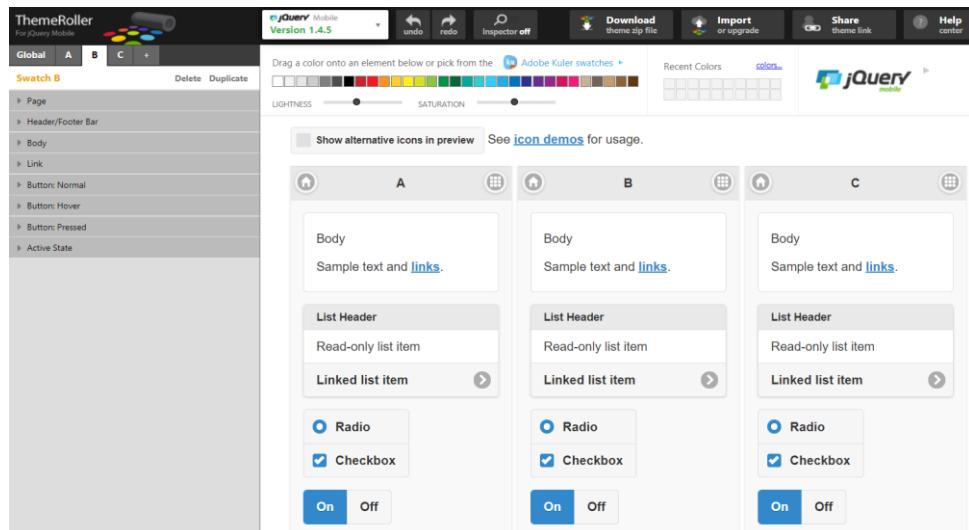
Şekil 13.19.'daki kodlar detaylı bir şekilde incelendiğinde metin kutuları haricindeki her bir standart form elemanın `"data-role"` parametresi ile jQuery Mobile'ın özel bir bileşenine dönüştürüldüğünü görüyoruz. Bu çalışma gerçek bir proje içinde hayata geçirirken tüm **form elemanlarının ve bileşenlerin** oluşturulacak **form etiketi** (`<form>`) içinde yer olması gerektiği unutulmamalıdır.

## jQuery Mobile TEMALARI

jQuery Mobile içerisinde Web sitemizin genel yapısı ile **görsel olarak uyum sağlayacak** farklı temaları oluşturabilme ve bu temalar arasında geçiş yapabilme

şansımız bulunmaktadır. **Temalar**, tüm jQuery Mobile bileşenlerinin rengini, gölgelendirme gibi efektleri ve yazı tiplerini değiştirebilmemizi sağlar. Bu çalışmalar jQuery Mobile ekibi tarafından geliştirilen ve bir tema oluşturucu olan **“Theme Roller”** ile yapılmaktadır.

**Theme Roller** Web tabanlı bir uygulamadır ve <https://theroller.jquerymobile.com/> adresinden erişilebilmektedir. Theme Roller'a ilk giriş yapıldığında bizi Şekil 13.20.'deki kullanıcı arayüzü karşılar.

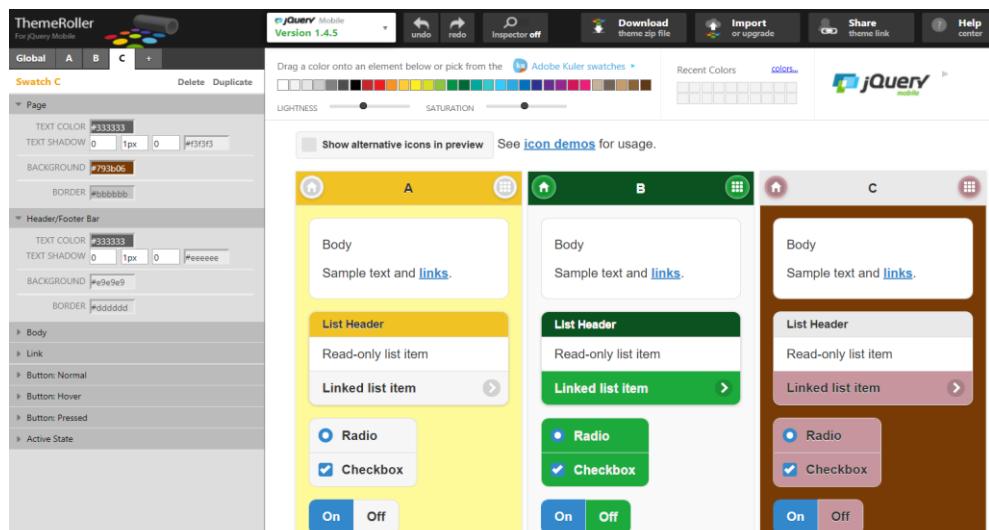


Şekil 13.20. Theme Roller Web uygulaması.

 Theme Roller, Web tabanlı arayüzü ile hızlı bir şekilde ilgi çekici temalar oluşturmanızı sağlar.

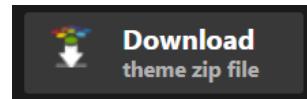
Theme Roller içerisinde varsayılan olarak **üç farklı tema** tanımlanabilir. Bu sayı arttırılabilir ya da azaltılabilir. Ekranın sol tarafında yer alan özellikler panelinde yapılan her değişiklik sağ bölümdeki temanın örnek görüntüsünde (ön izlemesi) **gerçek zamanlı** olarak görüntülenebilir.

Şekil 13.21.'de temalar üzerinde yapılan değişikliklerin örnek temalara yansımış hâlli gorüntülenmektedir. Her bir tema, alfabetin (Türkçe karakterler hariç) **sıradaki harfini** alarak isimlendirilir (a, b, c, d...) ve bu isim ile kod içinden çağrılarak kullanılır.



Şekil 13.21. Tema Değişikliklerinin Ön İzlenmesi.

Tema ile ilgili düzenlemeler tamamlandığı zaman tema, Theme Roller Web arayüzünün üst menüsündeki “*Download theme zip file*” (tema zip dosyasını indir) butonuna tıklanarak indirilir (Şekil 13.22.).



**Şekil 13.22.** Tema Tanımlamalarının Yer Aldığı Zip Dosyası İndirme Butonu.

İndirme butonuna tıklandığında Şekil 13.23.'teki iletişim kutusu görüntülenir. Bu iletişim kutusunda temanın adının belirleneceği bir metin kutusu sağ üst köşede yer almaktadır. İstedığınız bir ismi verebilirsiniz. Örnek çalışmamızda temamızın adı “yenitema” olsun.

  
Theme Roller'da oluşturduğunuz tema, zip dosyası olarak paketlenerek indirebilmeniz için sistem tarafından hazırlanır.

**Download Theme**

Theme Name

This will generate a Zip file that contains both a compressed (for production) and uncompressed (for editing) version of the theme.

To use your theme, add it together with the icon CSS file to the head of your page before the jquery.mobile.structure file, like this:

```
<!DOCTYPE html>
<html>
<head>

<title>jQuery Mobile page</title>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="css/themes/my-custom-theme.css" />
<link rel="stylesheet" href="css/themes/jquery.mobile.icons.min.css" />
<link rel="stylesheet" href="http://code.jquery.com/mobile/1.4.5/jquery.mobile.structure-1.4.5.min.css" />
<script src="http://code.jquery.com/jquery-1.11.1.min.js"></script>
<script src="http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.js"></script>

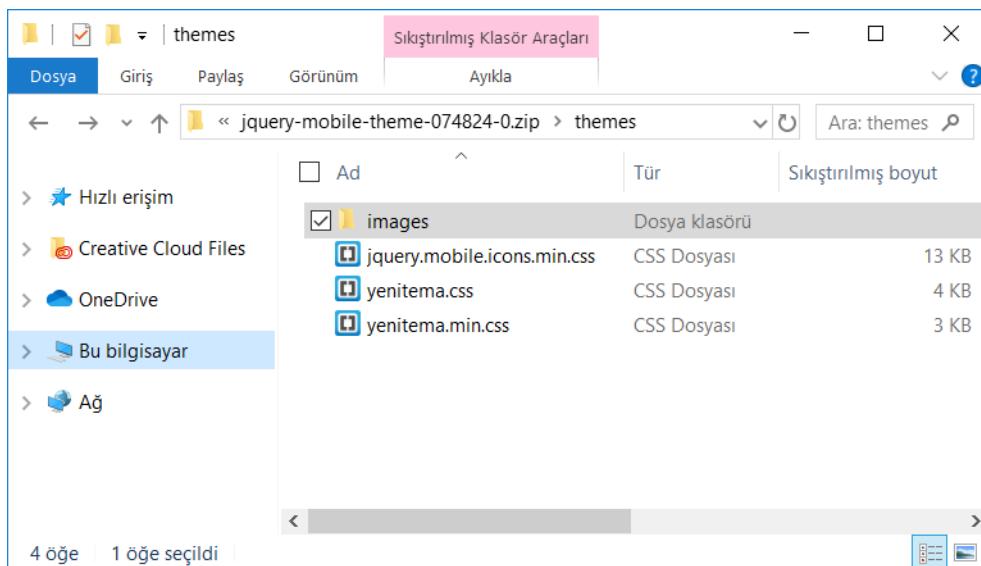
</head>
```

Tip: To edit your theme later, use the import feature to paste in the uncompressed theme file

**Close** **Download Zip**

**Şekil 13.23.** Tema İndirme İletişim Kutusu.

“Download Zip” (zip dosyasını indir) butonuna tıkladığınızda İnternet tarayıcınızın varsayılan dosya indirme lokasyonuna *zip dosyasını* indirecektir. Bu genelde “İndirilenler” adlı bir klasör olabilir. Zip uzantılı dosyayı bulunduğu konumda açalım ve içindeki dosyaları kopyalayarak proje klasörümüzün içine yapıştıralım. Zip dosyasının içeriği Şekil 13.24.'teki gibi olmalıdır.



**Şekil 13.24.** Zip Dosyasının İçeriği.

Zip dosyasının içindeki dosyaları incelediğimizde hem kendi oluşturduğumuz yeni temamızın normal ve sıkıştırılmış sürümlerinin yer almakta olduğunu (`yenitema.css` ve `yenitema.min.css`) hem de jQuery Mobile'ın ikon dosyaları ve tanımlamaları ile ilgili dosyaların yer aldığılığını görebiliriz.

Yeni oluşturulan bu CSS dosyalarını yeni temayı kullanacağımız dosyalarda ünitemin başındaki kurulum bölümünde olduğu gibi tanımlamalıyız. Tanımlama sonrası HTML dosyalarımızın başlık bölümü Şekil 13.25.'teki gibi olmalıdır.

```

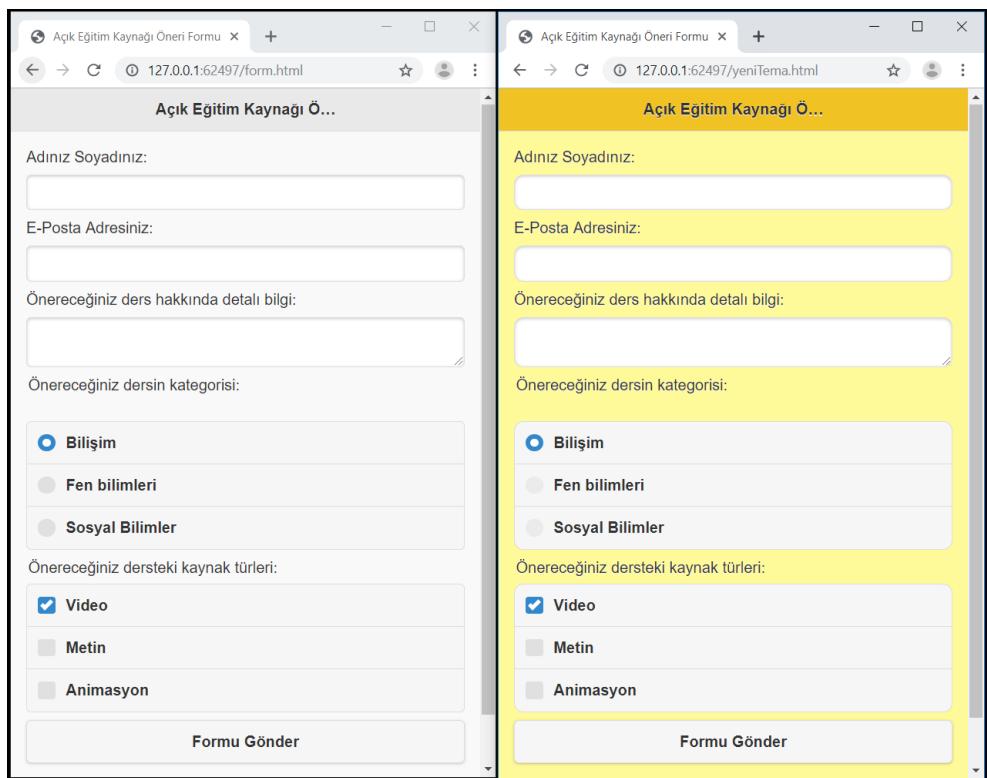
1. <head>
2. <meta charset="utf-8">
3. <title>jQuery Mobile Tek Sayfa</title>
4. <meta name="viewport" content="width=device-width,
 initial-scale=1">
5. <link rel="stylesheet" href="jquery.mobile.css" />
6. <link rel="stylesheet"
 href="jquery.mobile.icons.min.css" />
7. <link rel="stylesheet" href="yenitema.css" />
8. <script src="jquery.js"></script>
9. <script src="jquery.mobile.js"></script>
10. </head>
```

**Şekil 13.25.** Başlık Bölümünün Güncellenmiş Hâli.

Daha önce çalışma yaptığımız bir dosyaya **yenİ CSS dosyalarının** bağlantıları eklendiğinde sayfa içerisindeki bileşenler **yenİ temanın özelliklerini** alır. Şekil 13.26.'da daha önce standart tema kullandığımız Form örneğinin (`form.html`) önceki ve sonraki hâlleri görüntülenmektedir.



Oluşturduğunuz  
temanın projeniz içinde  
çalışabilmesi için zip  
paketin içerisindeki  
dosyaları proje  
klasörünüzü  
kopyalamalısınız.



Şekil 13.26. Standart Tema Ve Yeni Temanın Uygulanmış Hâli.

Theme Roller üzerinde tasarladığımız yeni jQuery Mobile temamız artık kullanılabilir durumda.



jQuery Mobile'da  
yaptığınız tüm  
çalışmaları mutlaka  
farklı tarayıcılarda da  
test edin.



### Bireysel Etkinlik

- Proje klasörünüz olan "jQueryMobile" içerisinde yeniTema.html adlı bir dosya oluşturun.
- Theme Rollar'da oluşturduğunuz yeni temanızın dosyalarını proje klasörünüzün içine kaydedin.
- Yeni temanızın CSS tanımlarını yeniTema.html dosyasının başlık bölümüne ekleyin.
- İstediğiniz jQuery Mobile bileşenini kullanın ve İnternet tarayıcınızda yeni temanın aktif olup olmadığını test edin.



## Özet

- jQuery Mobile, HTML5 temelli bir kullanıcı arayüz tasarım sistemidir.
- jQuery Mobile, sadece JavaScript dosyalarından oluşmaz, arayüz tasarımını sağlamak için CSS dosyaları da yer alır.
- jQuery Mobile, platform bağımsız olarak tanımlayabileceğimiz bir yapıda tasarlanmıştır. Bu yapısı birçok farklı İnternet tarayıcısında sorunsuz olarak çalışabilmesini sağlar.
- HTML belgesinde gezinme, olay işleme, animasyon ve diğer Ajax uygulamalarını çok daha basit hale getirir ve bu işlemlerin mobil uyumlu olarak gerçekleştirilemesini sağlar.
- jQuery Mobile, açık kaynak kodlu bir projedir.
- JavaScript kütüphanelerinin ve CSS tanımlamlarının genel çalışma prensipleri jQuery Mobile için de geçerlidir.
- jQuery Mobile kütüphane ve CSS dosyalarına iki farklı şekilde bağlantı sağlanabilir; dosya yerel bir adreste bulunabilir, İnternet üzerinde yer alan depolar üzerinden kullanılabilir.
- jQuery Mobile, kod geliştiriciler ve Web tasarımcılar için farklı kütüphane sürümleri sunar; bu sürümler arasındaki fark kodlar üzerinde değişiklik yapma kolaylığı sağlama ile ilgilidir.
- jQuery Mobile, Internet tarayıcılara bazı özel üst veri mesajları gönderir. Bu mesajlar tarayıcıların Web sitesini mobil cihazlara göre yorumlamasını sağlar.
- jQuery Mobile, Web sayfasının gövde bölümünde özel bir sayfa tanımlaması yapar, bu tanımlama mobil Web sitesine bir mobil uygulama görünümü verebilmek açısından büyük avantaj sağlar.
- İyi kurgulanarak oluşturulmuş bir şablon dosyası tüm proje için iyi bir başlangıç noktası ve altyapı sağlayacaktır.
- Tek bir HTML dokümanı içerisinde birden fazla Web sayfasının yer aldığı Web sitesi yapıları jQuery Mobile içinde rahatlıkla oluşturulur ve özel sayfa geçiş efektleri uygulanabilir.
- jQuery Mobile ile küçük cihazların ekranları için yer kazanımı sağlayacak açılır-kapanır paneller tasarlanabilir ve bu paneller farklı efektler ile kullanılabilir.
- Web sitelerinde veri gösterimi konusunda kullanılabilecek önemli bileşenlerden biri olan listelerin kullanımı jQuery Mobile içinde geliştirilere büyük bir avantaj sağlamaktadır.
- Listelerin kullanım özellikleri içinde bir filtreleme yapısı da bulunmaktadır. Bu yapı aynı zamanda basit bir arama fonksiyonu görevini de yerine getirir.
- Web sitelerinde veri gösterimi konusunda sıkça kullanılan bir başka bileşen ise tablo bileşenidir.
- Tablo bileşeninin fonksiyonelliği jQuery Mobile'in ek özellikler ile standart bir HTML tablosuna göre daha da artmıştır. Tablonun istenilen sütunları özel bir iletişim kutusu yardımıyla gizlenebilir.
- Web sitelerinde kullanıcı etkileşimi sağlayabilmek için kullanılan form elemanları, jQuery Mobile içerisinde özel bir arayüz tasarımına sahiptir.
- jQuery Mobile, farklı arayüz tasarımları geliştirebilmek için gelişmiş bir tema oluşturucusuna sahiptir. Theme Roller adındaki bu tema oluşturucu Web tabanlıdır ve tasarlanan tema ile ilgili ihtiyaç duyulan tüm dosyaları sıkıştırılmış bir paket ile geliştiricinin kullanımına sunar.

## DEĞERLENDİRME SORULARI

1. jQuery Mobile JavaScript ve CSS dosyalarının sıkıştırılmış sürümlerinin (min) sağladığı avantaj aşağıdakilerden hangisidir?
  - a) Boyut olarak daha küçük oldukları için Web sitesi optimizasyonu sağlar.
  - b) Dosyaların içeriğini herkes göremez, koruma sağlar.
  - c) Güncellemeye işlemleri daha kolay gerçekleştirilir.
  - d) Farklı sunuculara aktarılması kolaydır.
  - e) Daha çok fonksiyonu içinde barındırabilir.
2. İçerik Dağıtım Ağı (Content Delivery Network – CDN) ne işe yarar?
  - a) Aynı içeriğin farklı Web sitelerinde yayınlanabilmesini sağlar.
  - b) Web sitelerindeki görsel ve işitsel materyalleri depolar.
  - c) Günlük gelişmelerin İnternet üzerinden takip edilebileceği bir platform sunar.
  - d) İhtiyaç duyulan kaynak kodların İnternet üzerinde yer alan depolardan kullanılabilmesini sağlar.
  - e) jQuery Mobile'da oluşturulmuş içeriklerin yedeklenmesini sağlar.
3. jQuery Mobile çatısını oluşturan çekirdek dosyaları İnternet üzerindeki depolardan kullanmanın avantajı aşağıdakilerden hangisidir?
  - a) Dosyaların ilgili sürümlerinin her zaman en güncel hâlleri kullanılmış olur.
  - b) Dosya boyutları daha küçük olur.
  - c) Birden çok kütüphane dosyası ile çalışmaya olanak sağlar.
  - d) Yedekleme işlemlerini hızlandırır.
  - e) Yazılan kodların daha hızlı çalışması sağlanır.
4. jQuery Mobile gibi bir HTML5 çatısına arayüz tasarıımı açısından neden ihtiyaç duyulur?
  - a) Mobil Web sayfasının daha hızlı açılması için
  - b) Mobil cihaz kullanıcılarının iyi bir kullanıcı deneyimi yaşayabilmeleri için
  - c) Mobil Web sitelerinin genel güvenlik seviyelerinin arttırılması için
  - d) Donanım açısından düşük seviyeli mobil cihazların kullanılabilmesi için
  - e) İçerik olarak görsel ve işitsel materyallerin kullanılabilmesi için

5. jQuery Mobile için aşağıdakilerden hangisi söylenemez?
  - a) Açık kaynak kodlu bir projedir.
  - b) jQuery'nin JavaScript kütüphanelerini kullanır.
  - c) jQuery Mobile, masaüstü, dizüstü, tablet PC, cep telefonu gibi birçok farklı platform için arayüz tasarımları yapmamıza olanak verir.
  - d) jQuery Mobile sadece JavaScript kütüphanelerinden oluşur.
  - e) jQuery Mobile paketi JavaScript kütüphanelerine ek olarak CSS tanımlamalarını da içerir.
6. jQuery Mobile çoklu sayfa şablonunun sağladığı avantaj aşağıdakilerden hangisidir?
  - a) Web sitesinin toplam boyutunu azaltır.
  - b) Arama motorlarında görünebilirliği arttırmır.
  - c) Veri tabanı bağlantısını kolaylaştırır.
  - d) Web sayfasının hızlı yüklenmesini sağlar.
  - e) Tek bir HTML dosyasının içerisinde birden çok Web sayfasının barındırılmasına olanak sağlar.
7. Aşağıdakilerden hangisi jQuery Mobile sayfa geçiş efektlerinde bir iletişim penceresinin açılmasını sağlayan anahtar kelimedir (parametredir)?
  - a) Window
  - b) Page
  - c) Dialog
  - d) Flip
  - e) Flow
8. Aşağıdakilerden hangisi JQuery Mobile sayfa geçiş efektleri arasında yer almaz?
  - a) Soldurma (Fade)
  - b) Hareketlendirme (Animate)
  - c) Açılan pencere (Pop)
  - d) Dönerek (Turn)
  - e) Kayarak (Slide)
9. jQuery Mobile sayfa bileşenlerinden panelin arayüz tasarımı açısından sağladığı avantaj nedir?
  - a) Sayfanın daha hızlı yüklenmesini sağlar.
  - b) Çoklu sayfa tasarımına gereksinimi ortadan kaldırır.
  - c) Açıılır-kapanır bir yapıda olduğu için cihaz görünür alanında (ekranında) ayrıca bir yer kaplamaz.
  - d) Dokunmatik ekran kullanımına imkân sağlar.
  - e) Site içi aramayı kolaylaştırır.

10. jQuery Mobile ile tasarlanan liste öğelerinde arama fonksiyonunu yerine getiren basit sistemin adı nedir?
- a) Filtreleme
  - b) Sıralama
  - c) Doldurma
  - d) Eleme
  - e) Karşılaştırma

**Cevap Anahtarı**

1.a, 2.d, 3.a, 4.b, 5.d, 6.e, 7.c, 8.b, 9.c 10 -

## **YARARLANILAN KAYNAKLAR**

Baltalı, S. (2012). jQuery Mobile. İstanbul: KODLAB

JQuery Mobile Resmî Web Sitesi. 3 Temmuz 2019 tarihinde  
<https://jquerymobile.com/> adresinden erişildi.

JQuery Mobile Demo Sitesi. 3 Temmuz 2019 tarihinde  
<https://jquerymobile.com/demos> adresinden erişildi.

W3Schools. jQuery Mobile Ders Notları. 4 Temmuz 2019 tarihinde  
[https://www.w3schools.com/jquerymobile/tryit.asp?filename=tryjqmob\\_n\\_avbars](https://www.w3schools.com/jquerymobile/tryit.asp?filename=tryjqmob_n_avbars) adresinden erişildi.

# BOOTSTRAP İLE DUYARLI WEB TASARIMI



## İÇİNDEKİLER

- Bootstrap Nedir?
  - Bootstrap Paketi Neler İçerir?
  - Bootstrap Nasıl Çalışır?
- Bootstrap Kurulumu
- Bootstrap ile Duyarlı Web Sayfası Tasarımı
  - Duyarlı Web Tasarımı Kavramı
  - Duyarlı Web Sayfası Şablonu
- Bootstrap ile Web Sitesi İnşası
- Bootstrap Web Sitesi Örnekleri



## HEDEFLER

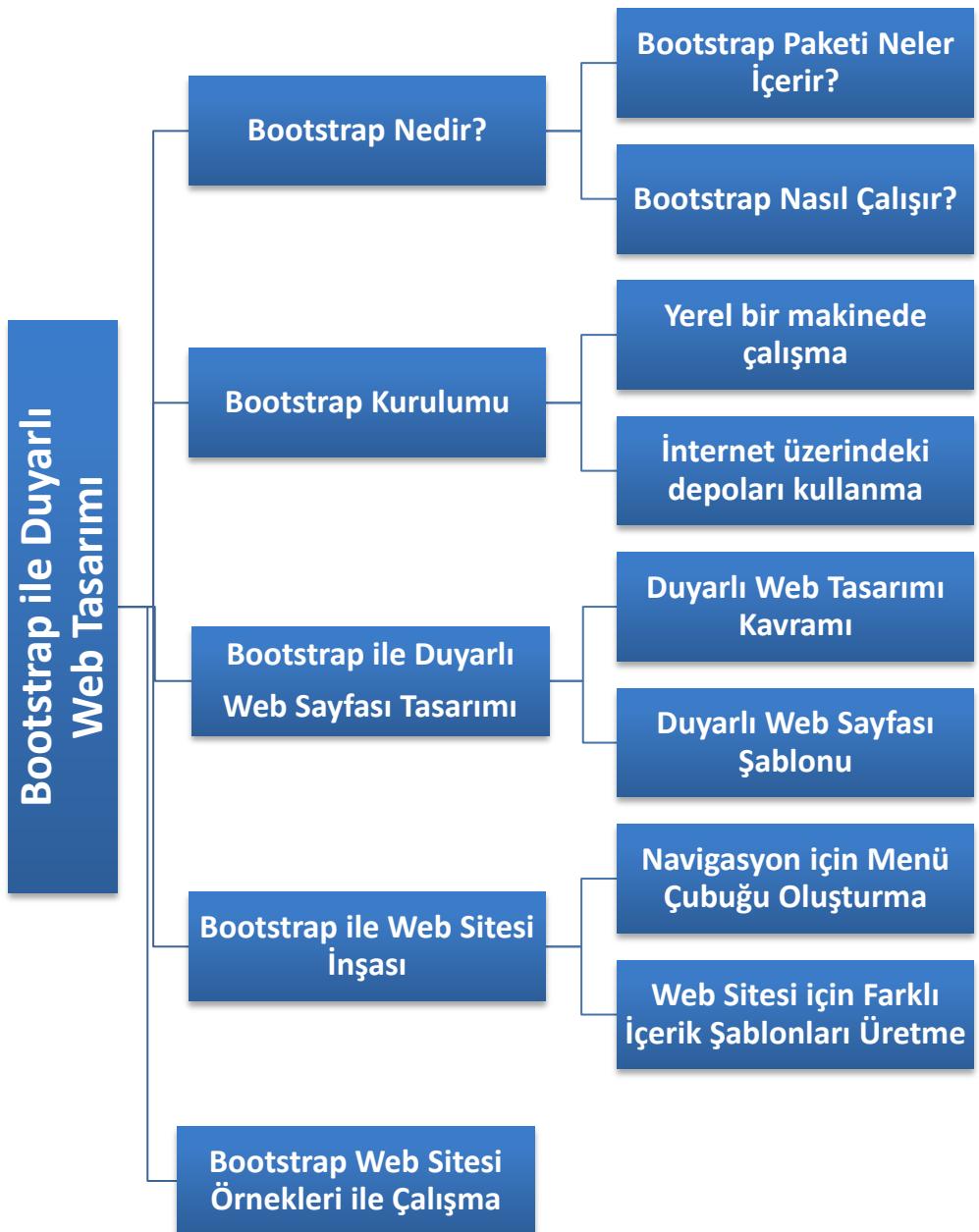
- Bu üniteyi çalıştıktan sonra;
  - Bootstrap çatısının kurulumunu yapabilecek,
  - Duyarlı Web tasarımı ile ilgili bilgi sahibi olacak,
  - Duyarlı Web sayfası şablonu oluşturabilecek,
  - Duyarlı Web sitesi inşa edebilecek,
  - Farklı Bootstrap örneklerini kendi Web sitesi çalışmalarınızda kullanabileceksiniz.



Atatürk Üniversitesi  
Açıköğretim Fakültesi

## İNTERNET PROGRAMCILIĞI II Öğr. Gör. Rafet Orçun MADRAN

ÜNİTE  
**14**



## GİRİŞ

İnternet üzerinden bilgiye erişen cihazların çeşitliliği, Web sitesi geliştiricilerini özellikle arayüz tasarımda farklı çözümler üretmeye sevk etmiştir. Mobil cihazların ilk akıllı sürümleri piyasada yer almaya başladığı zamanlarda Web sitelerinin masaüstü ve mobil sürümlerini iki ayrı site olarak tasarlamak sıkça kullanılan bir yöntemdi. Ancak bu yöntem mobil cihazların da kendi içinde çeşitlenmesi (farklı ekran boyutlarına sahip cep telefonları, tablet bilgisayarlar vb.), masaüstü sistemlerin farklı dizüstü modellerle geniş bir yelpazeye ulaşması ve artık akıllı TV gibi çok daha *büyük ekran boyutuna* sahip cihazların da oyuna girmesi ile ihtiyaçlara cevap veremez hâle geldi.

Çözüm, farklı ekran boyutuna sahip cihazların tamamı ile uyumlu çalışabilecek tek bir Web sitesi tasarılanmasında yatıyordu. *Duyarlı Web Tasarımı* (Responsive Web Design) olarak ifade edilen bu tasarım yaklaşımı, Web sitesinin görüntülendiği cihaza göre tasarımın kendini otomatik olarak *ekrana uyumlu* hâle getirmesi prensibine dayanıyordu. Bu sayede cihaz ekranının genişliği Internet tarayıcısı tarafından algılanmakta ve bu bilgiye göre tasarımını oluşturan bölümler *farklı şekilde ekrana yerleştirilerek* ya da *farklı şekilde boyutlandırılarak* görüntülenmektedir.

Duyarlı Web Tasarımı, gelişmiş Web tasarımlı bileşenlerine ihtiyaç duymaktadır. *HTML5* adını verdığımız bu tasarım bileşenleri *HTML*, *CSS* ve *JavaScript*'ten oluşur. Bu üç bileşenin birbiri ile uyumlu ve performanslı çalışabilmesi, geliştiricinin bilgi seviyesine bağlıdır. Aynı zamanda farklı Internet tarayıcıları ile de uyum içinde çalışmalıdır. Duyarlı Web Tasarımı'nı zorlu bir süreç dönüştüren bu teknik altyapı gereksinimlerini karşılayabilecek bütünlük sistemlere *çatı* (framework) adı verilir. *Çatılar*, geliştiricinin tasarım sürecinde ihtiyacı olan tüm karmaşık tanımlamaların önceden yapılandırıldığı hazır bir geliştirme ortamı sunar. Bu ortamı kullanan geliştiriciler, teknik detaylarda boğulmayarak tasarımını *çok daha hızlı* şekilde hayatı geçirebilirler.



Bootstrap,  
Duyarlı Web Siteleri  
geliştirebileceğimiz  
HTML5 tabanlı bir  
çatıdır (framework).

Bu üitede, HTML5 çatıları arasında en çok tercih edilen *Bootstrap* adlı çatı üzerinde durulacaktır. Bootstrap'in temel özellikleri, geliştiriciye sağladığı imkânlar, kurulumdan başlayarak farklı örneklerin kullanımına kadar bu ünite içerisinde detaylandırılacaktır.

## BOOTSTRAP NEDİR?

Bootstrap, *mobil cihazlar* için öncelikli ve *duyarlı web siteleri* geliştirmek için arayüz bileşen kütüphanelerine sahip bir çatıdır. Açık kaynak kodlu bir araç olan Bootstrap, projeler için hızlı bir şekilde *prototip* oluşturulabilmesi, ızgara sisteminin kullanımı ve jQuery üzerine inşa edilmiş güçlü bileşenleri ile popüler bir Web çatısıdır.

Bootstrap ile ilgili ihtiyaç duyulabilecek tüm kaynak dosyalar ve dokümantasyon <https://getbootstrap.com/> adresinde yer almaktadır (Şekil 14.1.).



# Bootstrap

Build responsive, mobile-first projects on the web with the world's most popular front-end component library.

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery.

[Get started](#) [Download](#)

Currently v4.3.1



14.1. Bootstrap Web sitesi.

## Bootstrap Paketi Neler İçerir?

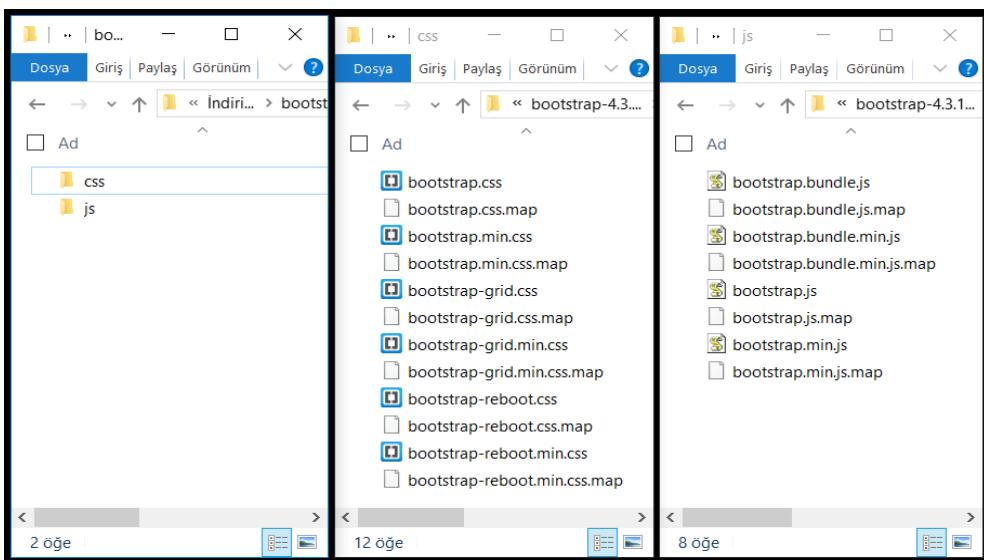
Bootstrap paketinin içeriğini temel olarak iki farklı dosya türü oluşturur. Bu dosya türleri *CSS tanımlamalarının* yer aldığı \*.css uzantılı dosyalar ve *JavaScript kütüphanelerinin* yer aldığı \*.js uzantılı dosyalardır. Birçok ek bileşenin de kullanılabildiği Bootstrap'in çekirdek dosyalarını Şekil 14.2.'deki indirme sayfasından (<https://getbootstrap.com/docs/4.3/getting-started/download/>) kendi kişisel bilgisayarınıza indirebilirsiniz.

  
Bootstrap paketi,  
CSS tanımlamalarından  
ve JavaScript  
kütüphanelerinden  
oluşmaktadır.

The screenshot shows the Bootstrap download page. It features a sidebar on the left with links for Getting started, Layout, Content, Components, Utilities, Extend, Migration, and About. The main content area has a 'Download' section with a 'Download' button and a note about a limited-time offer for Adobe Stock images. Below it is a 'Compiled CSS and JS' section with a 'Download' button and a note about including minified CSS and JS bundles. A sidebar on the right lists package managers: npm, yarn, RubyGems, Composer, and NuGet.

Şekil 14.2. Bootstrap İndirme Sayfası.

İndirme işlemi gerçekleştirgiinde Bootstrap dağıtım sürümü sıkıştırılmış dosya formatında (zip) bilgisayarınıza kaydedilmiş olur. Zip dosyası açıldığında ana klasörün içinde iki farklı alt klasör yer alır. Bu alt klasörler “*css*” ve “*js*” klasörleridir. Klasörlerin içeriği Şekil 14.3.'te görüntülenmektedir.



Şekil 14.3. Bootstrap Dağıtım Sürümü Klasör İçerikleri.

Daha önceki ünitelerde paket içeriklerinden bahsettiğimizde her bir dosyanın iki farklı sürümü olduğunu belirtmiştim. Bu dosya sürümleri birbirlerinden “min” eki ile ayrılmaktaydı. Dosyaların uzantısında yer alan “min” eki o dosyanın *sıkıştırılmış* bir yapıda hazırlandığını belirtiyordu. Dosya boyutu olarak *daha az yer kaplayan* “min” sürümlerini projelerde kullanmak Web sitesinin optimizasyonu açısından avantaj sağlıyordu.

## Bootstrap Nasıl Çalışır?

Bootstrap’ın çalışma mantığı Ünite 12’deki jQuery kütüphanesinin ve Ünite 13’teki jQuery Mobile çatısının çalışma mantığı ile paralellik göstermektedir. Çatının kullanılacağı HTML dosyasından ilgili dosyalara (css ve js) *referans verilerek* bağlantı sağlanması Bootstrap çatısının aktif hâle gelmesi için yeterlidir. Referans verilme işlemi gerçekleştirildikten sonra JavaScript kütüphanesindeki fonksiyonları ve CSS tanımlamalarını *kod içinden çağırarak* istenilen yapı inşa edilebilir. Şekil 14.4.’te standart bir Bootstrap sayfa şablonu yer almaktadır.



Bootstrap, diğer Web geliştirme çatılarına benzer bir çalışma mantığına sahiptir.

```

1. <!doctype html>
2. <html lang="tr">
3. <head>
4. <meta charset="utf-8">
5. <meta name="viewport" content="width=device-width,
initial-scale=1">
6. <link rel="stylesheet" href="bootstrap.css">
7. <script src="jquery.js"></script>
8. <script src="popper.js"></script>
9. <script src="bootstrap.js"></script>
10. <title>Bootstrap</title>
11. </head>
12. <body>
13. <h1>Bootstrap Çatı Şablonu</h1>
14. <p>Web sayfasının içeriği burada yer alacak!</p>
15. </body>
16. </html>
```

Şekil 14.4. Bootstrap Sayfa Şablonu Örnek Kodları.

## BOOTSTRAP KURULUMU



Bootstrap kurulumu iki farklı şekilde gerçekleştirilebilir: internet üzerinde yer alan depolardan, bilgisayarınızın sabit diskine kaydederek.

Bootstrap kurulumu iki farklı şekilde gerçekleştirilebilir. Bunlardan ilki Bootstrap çekirdek dosyalarını *internet üzerinde yer alan depolardan* kullanmaktadır. Bu yöntemin avantajı ilgili sürümlerin en güncel hallerine sürekli olarak erişebilmemizdir. Dezavantaj ise geliştirme ve yayın aşamasında *internet bağlantısına* olan ihtiyaçtır. Standart özelliklere sahip bir Bootstrap projesinde ihtiyaç duyulan çekirdek dosyalar eğer internet üzerinden kullanılacaksa Şekil 14.5.'teki satırların Web sayfasının başlık (<head>) bölümüne yerleştirilmesi yeterli olacaktır.

```

1. <link rel="stylesheet"
 href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.
 1/css/bootstrap.min.css">
2. <script src="https://code.jquery.com/jquery-
 3.3.1.slim.min.js"></script>
3. <script
 src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1
 .14.7/umd/popper.min.js"></script>
4. <script
 src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1
 /js/bootstrap.min.js"></script>
```

**Şekil 14.5.** Bootstrap Çekirdek Dosyalarının Internet Üzerindeki Depolardan Kullanımı.

Diğer yöntem ise Şekil 14.5.'te referans verilen dosyaları *bilgisayarınızın sabit diskine kaydederek* bağlantı kurmaktır. Bu yöntem, internet bağlantınız olmadan da geliştirme yapmanıza olanak tanır ancak ilgili sürümlerde bir güncelleme olduğunda bu *güncellemeden faydalamanızı engeller*.

Kurulumu, ihtiyacımız olan dosyaları sabit diskimizde oluşturduğumuz bir proje klasörünün içine kaydederek tamamlayalım ve bir şablon dosya oluşturarak Bootstrap projemizi test edelim.



Web sitesi tasarılığında aynı türdeki dosyaların alt klasörlerde toplanması geliştirme sürecine olumlu katkı sağlar.

Web siteleri yaşayan organizmalara benzediği için zamanla dosya sayılarında çok ciddi bir artış olabilir. Bu açıdan ana proje klasörümüz altındaki dosyaları da alt klasörlere ayırarak daha düzenli bir yapı oluşturabiliriz. Genelde CSS tanımlamalarının yer aldığı \*.css uzantılı dosyalar “css” adlı bir klasör içinde, JavaScript kütüphanelerinin yer aldığı \*.js uzantılı dosyalar da “js” klasörü içinde yer alır. Proje klasörümüz ve içeriği Tablo 14.1.'deki gibi olmalıdır.

Bootstrap'in *sadece CSS* özelliklerinden faydalanimak isteniyorsa *JavaScript kütüphaneleri kullanılmayabilir*. Ancak bu durumda sayfa içindeki birçok etkileşim ve JavaScript ile gerçekleştirilecek fonksiyonlar devre dışı kalacaktır.

**Tablo 14.1.** Bootstrap Örnek Proje Klasör Ve Dosya Yapılanması.

Proje Ana Klasörü	Alt Klasörler	Dosya İsimleri	Dosya Açıklama
bootstrap	css	bootstrap.css	Bootstrap CSS tanımlama dosyası
	js	bootstrap.js	Bootstrap JavaScript Kütüphanesi
		jquery.js	jQuery JavaScript Kütüphanesi
		popper.js	Popper JavaScript Kütüphanesi



### Bireysel Etkinlik

- Sabit diskinizde "bootstrap" adlı bir proje klasörü oluşturun.
- Proje klasörü içerisinde "css" ve "js" adlarında alt klasörler oluşturun.
- Şekil 14.5.'teki adreslerden CSS ve JS dosyalarını bilgisayarınıza indirerek, Tablo 14.1.'deki gibi isimlendirin ve ilgili alt klasörlere yerleştirin.
- Şekil 14.4.'teki örnek kodlardan faydalananarak bir şablon dokümanı oluşturun ve İntenet tarayıcınızda test edin.
- CSS ve JS dosyalarına referans verirken bulundukları klasörleri dikkate almayı unutmayın.



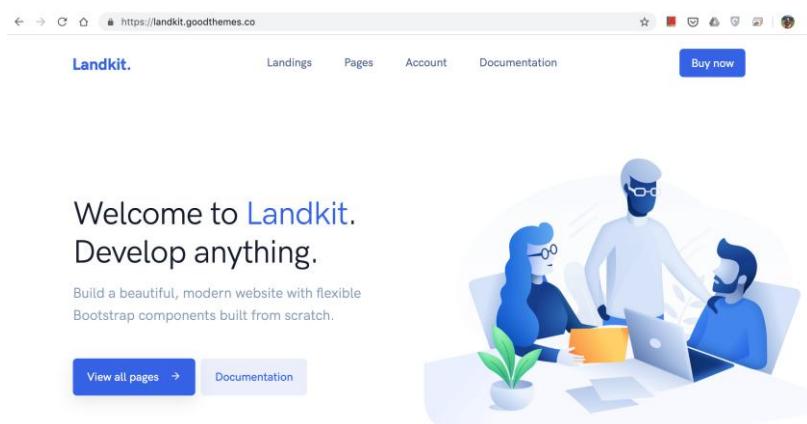
Bootstrap çatısının en önemli özelliği farklı ekran boyutlarına sahip cihazlar için duyarlı tasarımların yapılabilmesidir.

## BOOTSTRAP İLE DUYARLI WEB SAYFASI TASARIMI

Bootstrap'in en önemli özelliklerinden birinin duyarlı web sayfaları inşa edebilmek olduğundan bahsetmiştik. Duyarlı bir web sayfasının temel özelliği olan *ekran boyutuna göre sayfa öğelerinin yeniden yerleşimi* ile ilgili bir örnek çalışma yapalım. Bu örnek çalışma aynı zamanda Web sitemizde kullanacağımız temel sayfa yapısının da şablonu olsun.

### Duyarlı Web Tasarımı Kavramı

Bir şablon oluşturmaya başlamadan önce duyarlı web sitesinden ne kastettiğimizi bir örnek ile netlestirelim. Şekil 14.6.'da bir masaüstü ya da dizüstü bilgisayar tarafından görüntülenen Bootstrap ile tasarlanmış bir Web sitesinin ekran görüntüsü yer almaktır.



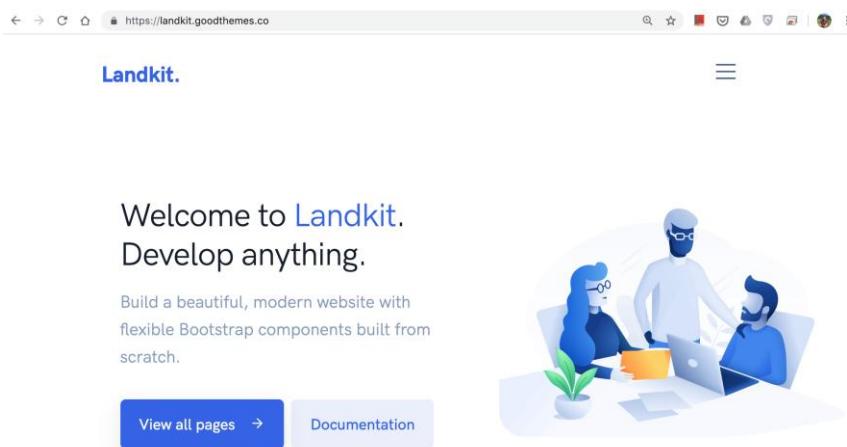
Şekil 14.6. Örnek Web Sitesi Masaüstü / Dizüstü Bilgisayar İle Görüntüleme.

Şekil 14.6.'daki ekran görüntüsünü incelediğimizde *sayfa içi navigasyonu* sağlayan menünün sağ üst bölümde konumlandığıını görüyoruz. Navigasyondan sonra sayfanın 2 bölüme ayrıldığını, sol bölümde slogan, metin ve

butonlarını, sağ bölümde ise bir görselin yer aldığıni görüyoruz.

Web sayfalarının farklı ekran boyutlarına göre test edebilmenin çeşitli yolları vardır. Bunlardan biri İnternet tarayıcınızın *pencere genişliğini daraltarak*, sanki daha küçük bir ekranda görüntülenmiş gibi yapmaktır. Diğer bir yöntem ise İnternet tarayıcınızın *yakınlaştırma* (zoom) özelliğinden %100'ün üzerinde bir değer ayarlayarak içeriğin oran olarak büyümeyi sağlamaktır. Eğer farklı cihazlarda (cep telefonu, tablet PC vb.) fiziksel olarak test imkânı bulunmuyorsa, yukarıda bahsedilen yollar denenebilir ve çok büyük oranda test ortamı size gerçege yakın bir fikir verecektir.

Şekil 14.7.'de İnternet tarayıcımızın pencere genişliğini biraz azaltıyoruz (ya da yakınlaştırma değerini örn. %125 olarak belirliyoruz) ve sayfa tasarımındaki değişiklikleri gözlemliyoruz.



Şekil 14.7. Örnek Web Sitesi Tablet PC (Temsili Olarak) İle Görüntüleme.

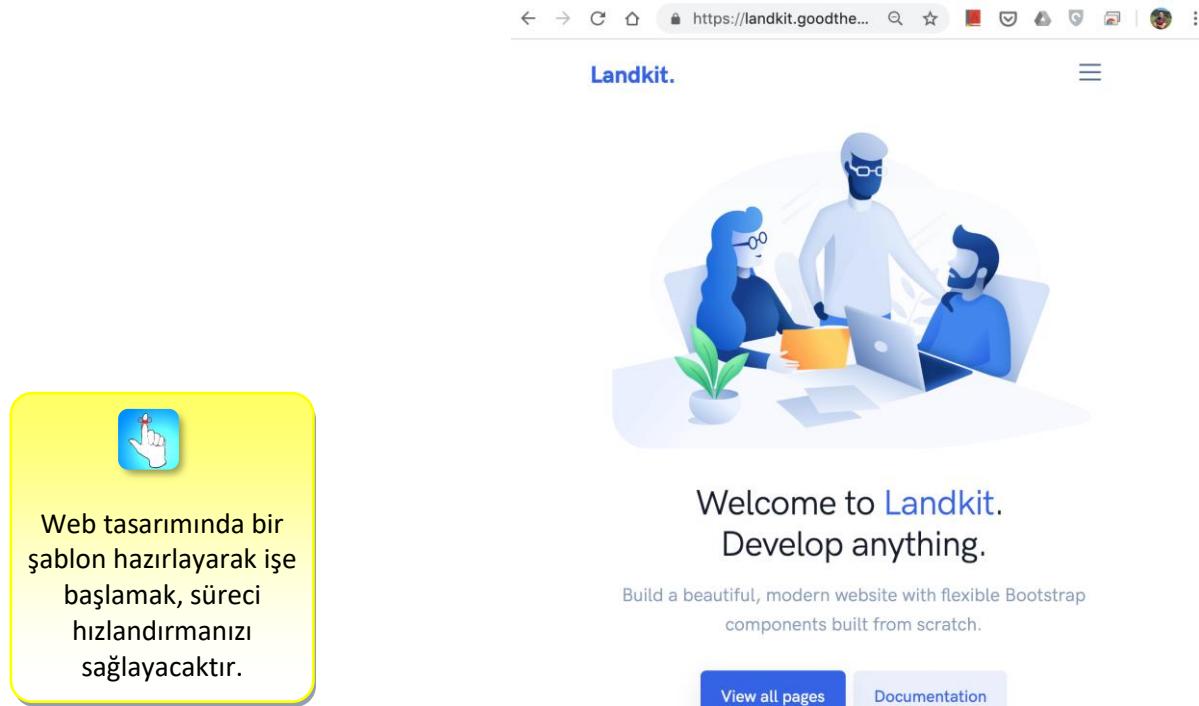
Şekil 14.7.'yi dikkatlice incelediğimizde sağ üstte yer alan navigasyon bölümünün değişikliğe uğradığını görüyoruz. Şekil 14.6.'da menü öğeleri açık bir şekilde duruyorken, Şekil 14.7.'de menü öğeleri artık bir butonun (*alt alta üç çizgi*) altına gizlenmiş durumda. Ekran boyutu azaldıkça sayfa içerisinde yer alan öğelerin daha az yer kaplaması amaçlanıyor.

Bir sonraki aşamayı gözlemleyebilmek için ekran boyutunu iyice küçültmemiz ya da içeriğin oranını yapay bir şekilde (yakınlaştırma kullanarak) daha da büyütmemiz gerekiyor.

Şekil 14.8.'de artık sayfayı bir *cep telefonundan* görüntüleyipmiş gibi düşününebiliriz.

 Geliştirme ortamınızda farklı cihaz ekranlarını test edebilmek için İnternet tarayıcınızın yakınlaştırma özelliğinden faydalana bilirisiniz.

 Internet tarayıcınızın pencere boyutlarını değiştirerek farklı cihaz ekranlarını test edebilirisiniz.



Şekil 14.8. Örnek Web Sitesi Cep Telefonu (Temsili Olarak) İle Görüntüleme.

Şekil 14.8.'i incelediğimizde sayfa içerisindeki öğelerin *yeni yerleşimleri* hemen kendini belli etmekte. Ekranın genişliği daraldığı için artık slogan bölümü ve görsel öğe *yan yana* iki sütunda duramıyor. Bu öğeler *alt alta* sıralanmış durumda.

Yukarıda 3 farklı ekran boyutuna göre test ettiğimiz Web sayfası duyarlılık ile ilgili kavramı anlayabilmemiz için güzel bir örnek oluşturdu. Web sayfasının geliştiricisi aslında *tek bir sayfa tasarladı* ancak bu sayfa *üç farklı ekran* boyutuna göre kendini adapte edebildi ve uyum sağladı.



#### Bireysel Etkinlik

- Internet tarayıcınız ile <https://themes.getbootstrap.com/> adresini ziyaret edin.
- Bu adreste yer alan temaları tarayıcınızda farklı ekran boyutlarına göre görüntüleyin.
- Farklı ekran boyutlarını tarayıcınızın pencere genişliğini daraltarak ya da yakınlaştırmayı arttırarak simülle edebilirsiniz.

## Duyarlı Web Sayfası Şablonu

Önceki bölümde “Duyarlı Web Tasarımı” kavramını örneklerle incelediğimize göre artık kendi duyarlı sayfa şablonumuz üzerinde çalışmaya başlayabiliriz. Kurulumda, ihtiyacımız olan çekirdek dosyaları temin etmişik ve bunları proje klasörümüz içindeki alt klasörlere yerleştirmiştik.

## Sayfanın başlık kısmı

Şimdi proje klasörümüz içerisinde “*sablon.html*” adlı bir dosya oluşturalım. Bu dosyanın *başlık* (*<head>*) bölümü Şekil 14.8.’deki gibi kodlanmalıdır. *Bu bölüm* bizim Bootstrap çalışmalarımızda (ek bir JS kütüphanesi ya da bir CSS tanımlaması kullanılmadıkça) standart olarak *her zaman yer alacaktır*.

```
1. <!doctype html>
2. <html lang="tr">
3. <head>
4. <meta charset="utf-8">
5. <meta name="viewport" content="width=device-width,
 initial-scale=1">
6. <link rel="stylesheet" href="css/bootstrap.css">
7. <script src="js/jquery.js"></script>
8. <script src="js/popper.js"></script>
9. <script src="js/bootstrap.js"></script>
10. <title>Bootstrap Şablon Sayfası</title>
11. </head>
12. <body>
13. </body>
14. </html>
```

**Şekil 14.8.** Bootstrap Şablon Sayfası Başlık Bölümü.

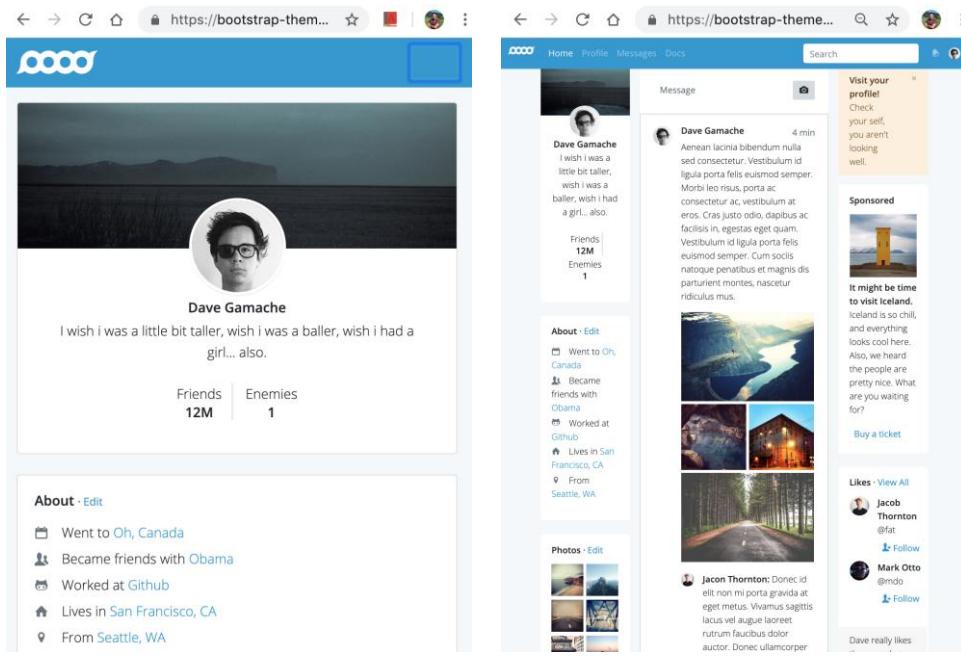
Şekil 14.8.’deki başlık bölümü kodlarında satır 6, 7, 8 ve 9 Bootstrap çatısını oluşturan çekirdek dosyalara verilen *referanslardır*. Bu dosyaların bağlantılarının doğru bir şekilde verilmesi çok önemlidir. Bulundukları klasör adı ve dosya adları tam olarak yazılmalı ve bağlantının kırık olmadığı mutlaka test edilmelidir. Başlık bölümündeki bir diğer önemli satır ise 5. satırdır. 5. satırda görülebilir alanla ilgili bir üst veri tanımlaması (viewport / görüntü alanı) yer almaktadır. Bu tanımlama, Web sayfasının duyarlı olarak tasarılandığını ve görülebilir alanın küçük olmasından dolayı internet tarayıcısının *siğdırma işlemi yapmamasını* söyler.

“Viewport” üst veri tanımlamasını alan tarayıcı siğdırma işlemi yapmaz, sayfayı küçüterek okunması zor bir hale sokmaz, duyarlı tasarımın öğeleri yeniden sayfa içinde konumlandırmasına ve yapılandırmasına (yazı tipi boyutu vb.) izin verir.

Şekil 14.9.’da “*viewport*” üst veri tanımlaması *yapılmış* ve *yapılmamış* örnekler yan yana yer almaktadır. Örnekler arasındaki farklılık “viewport” tanımlamasının duyarlı Web tasarımındaki önemini gözler önüne sermektedir.



Duyarlı Web Tasarımı, Internet tarayıcılarına tasarımın genel özellikleri ile ilgili üst veri tanımlamaları da sağlar.



**Şekil 14.9.** Aynı Web Sitesinin Temsili Mobil Cihaz Ekranında Görüntülenmesi; "Viewport" Tanımlaması Yapılmış (Solda) Ve Yapılmamış (Sağda) Web Sayfası Örnekleri.

### Sayfanın gövde kısmı

Duyarlı sayfa şablonumuzun başlık (`<head>`) bölümü ile ilgili düzenlemeleri tamamladığımıza göre artık **gövde** (`<body>`) bölümü ile ilgili olan çalışmalara başlayabiliriz.

Bootstrap sayfa yapısının en önemli bileşeni **bölmelerdir** (`<div>`). Bölmeler bize **izgara** (grid) adı verilen **hayali bir şablon** üzerinde çalışmamıza olanak sağlar. Bu sayede HTML içerisinde istediğimiz sayfa yerleşimini gerçekleştirebiliriz.

Sayfamızın gövde bölümünü oluşturacak kodlar Şekil 14.10.'daki gibi olmalıdır.

```

1. <body>
2. <div class="jumbotron text-center">
3. <h1>Bootstrap Sayfa Şablonu</h1>
4. <p>İnternet tarayıcısının pencere genişliğini
5. değiştirerek sayfada yaşanan değişiklikleri
6. gözlemleyebilirsiniz!</p>
7. </div>
8. <div class="container">
9. <div class="row">
10. <div class="col-sm-4">
11. <h3>Başlık</h3>
12. <p>İçerik</p>
13. </div>
14. <div class="col-sm-4">
15. <h3>Başlık</h3>
16. <p>İçerik</p>
17. </div>
18. <div class="col-sm-4">
19. <h3>Başlık</h3>
20. <p>İçerik</p>
21. </div>

```

```

20. </div>
21. </div>
22. </body>

```

**Şekil 14.10.** Bootstrap Şablon Sayfası Gövde Bölümü.

HTML5 çatıları, sayfa içinde öğelerin yerlesimi için izgara sistemi ve boyutlandırılabilir sütunlar sağlar.

Şekil 14.10.'daki kod yapısını incelediğimizde gövde bölümünde *iki ana bölmeyi* (`<div>`) yer aldığığini görüyoruz. Bu bölmelerden sayfanın üst bölümünde yer alacak karşılama mesajının olduğu bölümme satır 2-5 arasında, sütunların olduğu içerik bölmesi ise satır 6-21 arasına kodlanmış durumda. 2. satırda yer alan CSS sınıf tanımlaması (`class="jumbotron text-center"`) bu bölümdeki metinlerin sayfa içinde ortalanmasını sağlamakta. 6. satırda sınıf tanımlaması (`class="container"`) ise sütunlar için bir ana taşıyıcı (konteynır) görevi görmekte.

Bootstrap içerisinde sayfa içi yerleşiminde kullanabileceğimiz *standart* olarak *12 sütunlu* bir yapı bulunmaktadır. Bu sütunlar bölmelerin (`<div>`) içerisinde *CSS sınıfları* (`class`) olarak tanımlanabilmekte ve her bölmeyi kaç sütundan oluşacağı belirtilmektedir. Şekil 14.10.'da 8., 12. ve 16. satırlarda tanımlanan bölüm sınıfı (`<div class="col-sm-4">`) sayfanın *üç eşit parçaya* bölünmesini sağlamaktadır.

Burada önemli olan nokta toplam sütun sayısının 12'ye tamamlanmış olmasıdır. Örnek olarak *ilk sütun dar*, ikinci ve üçüncü sütunlar daha geniş ve *birbirine eşit* olsun isteniyorsa uygulanacak formül *2-5-5* olabilir. İlk bölüm (`<div>`) 2 sütundan, 2. ve 3. bölmeler ise 5 sütundan oluşacaktır.

Şekil 14.10.'daki kodları "sablon.html" dosyasına eklediğimizde ve İnternet tarayıcımızda dosyayı açtığımızda ekran görüntüsü Şekil 14.11.'deki gibi olacaktır.



#### Bootstrap Nedir?

Bootstrap, mobil cihazlar için öncelikli ve duyarlı web siteleri geliştirmek için arayüz bileşen kütüphanelerine sahip bir çatıdır.

Açık kaynak kodlu bir araç olan Bootstrap, projelerin hızlı bir şekilde prototiplenmesini, izgara sisteminin kullanımı ve jQuery üzerine inşa edilmiş güçlü bileşenleri ile popüler bir Web çatısıdır.

#### Bootstrap Paketi

Bootstrap paketinin içeriğini temel olarak iki farklı dosya türü oluşturur. Bu dosya türleri CSS tanımlamalarının yer aldığı \*.css uzantılı dosyalar ve JavaScript kütüphanelerinin yer aldığı \*.js uzantılı dosyalarıdır.

Birçok ek bileşenin de kullanılabilmesi Bootstrap'in çekirdek dosyalarını getbootstrap.com adresinden kendi kişisel bilgisayarınıza indirebilirsiniz.

#### Bootstrap Kurulumu

Bootstrap kurulumu iki farklı şekilde gerçekleştirilebilir. Bunlardan ilki Bootstrap çekirdek dosyalarını İnternet üzerinde yer alan depolardan kullanmaktadır.

Bu yöntemin avantajı ilgili sürümlerin en güncel hallerine sürekli olarak erişebilmenizdir. Dezavantajı ise geliştirme ve yayın aşamasında İnternet bağlantısına olan ihtiyaçtır.

**Şekil 14.11.** Bootstrap Şablon Dosyası Ekran Görüntüsü.

Şekil 14.11.'de ilk duyarlı web sayfası tasarımımızı gerçekleştirmiştir. Bu görünümden tasarımımızın standart bir *masaüstü bilgisayar* için iyi bir sayfa yerleşimi sunduğunu anlayabiliyoruz. Şimdi İnternet tarayıcımızın penceresini *genişlik olarak daraltalım* ve sonucu gözlemyelim (Şekil 14.12.).



Duyarlı Web Sayfaları, görüntünlendikleri ekrana göre öğelerin sayfa içi yerleşimlerini yeniden yaparlar.

## Bootstrap Nedir?

Bootstrap, mobil cihazlar için öncelikli ve duyarlı web siteleri geliştirmek için arayüz bileşen kütüphanelerine sahip bir çatıdır.

Açık kaynak kodlu bir araç olan Bootstrap, projelerin hızlı bir şekilde prototiplenmesini, ızgarası sisteminin kullanımı ve jQuery üzerine inşa edilmiş güçlü bileşenleri ile popüler bir Web çatısıdır.

## Bootstrap Paketi

Bootstrap paketinin içeriğini temel olarak iki farklı dosya

**Şekil 14.13.** Bootstrap Şablon Dosyası Mobil Cihaz (Temsili) Ekran Görüntüsü.

Şekil 14.13.'teki ekran görüntüsünü incelediğimizde 3 sütundan oluşan içerik alanının *kendini yeniden yapılandırdığını* ve *tek sütunlu* bir yapıya dönüştüğünü gözlemliyoruz. Tasarımımız farklı boyutta görüntü alanına (viewport) sahip cihazlar için de iyi bir çözüm üretmiş durumda.



### Bireysel Etkinlik

- Proje klasörünüzde "sablon.html" adlı bir dosya oluşturun.
- Şekil 14.8. ve Şekil 14.10.'daki kod yapılarından faydalananarak "sablon.html" içerisinde bir örnek sayfa kodlayın.
- Kodladığınız sayfayı Internet tarayıcınızda farklı ekran genişliklerinde test edin ve değişiklikleri izleyin.

## Bootstrap ile Web Sitesi İnşası

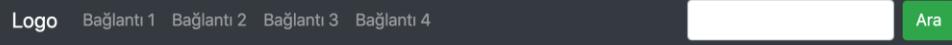


Duyarlı Web Tasarımında, navigasyonu sağlayacak menü ögesinin de duyarlı bir yapıda tasarlanması gereklidir.

Duyarlı bir sayfa şablonunu başarıyla oluşturduğumuza göre, bu deneyimimizi bir Web sitesinin inşa sürecinde kullanabiliriz. Navigasyonu sağlayacak *menü yapısını da duyarlı bir şekilde oluşturup* bunu mevcut yapıya ekleyerek 4-5 sayfadan oluşan bir Web sitesi hayata geçirelim.

Web sitesinin navigasyonu için tasarlayacağımız menü çubuğumuzda en solda bir *logo* alanı, daha sonra siteyi oluşturan *sayfalara bağlantıları* yer alacağı bir bölüm ve en sağda ise bir *arama bölümü* yer alsın. Bu menü çubuğu ekranın görülebilir alanına göre belirli bir genişliğin altına düşündüğü zaman tek bir menü düğmesi altında toplanacak bir yapıda olsun. Duyarlı menü çubuğumuzun geri

görüntüsü şekil 14.14.'teki gibi olacaktır.



**Şekil 14.14.** Duyarlı Menü Çubuğu Genel Görünümü.

Şekil 14.14.'te menü çubuğunu oluşturan kod yapısı Şekil 14.15.'te yer almaktadır. Menü çubuğunun içinde yer alan öğeler temelinde **HTML liste elemanlarıdır**. Çok farklı şekillerde menü çubukları tasarlamak mümkündür.



Menü çubuğu  
oluşturmadıkta kullanılan  
en önemli öğe HTML  
liste elemanlarıdır.

```

1. <nav class="navbar navbar-expand-lg navbar-dark bg-dark" style="margin:24px 0;">
2. Logo
3. <button class="navbar-toggler navbar-toggler-right" type="button" data-toggle="collapse" data-target="#navb">
4.
5. </button>
6. <div class="collapse navbar-collapse" id="navb">
7. <ul class="navbar-nav mr-auto">
8. <li class="nav-item">
9. Bağlantı 1
10.
11. <li class="nav-item">
12. Bağlantı 2
13.
14. <li class="nav-item">
15. Bağlantı 3
16.
17. <li class="nav-item">
18. Bağlantı 4
19.
20.
21. <form class="form-inline my-2 my-lg-0" method="get" target="_blank" action="http://www.google.com/search">
22. <input class="form-control mr-sm-2" type="text" name="q">
23. <button class="btn btn-success my-2 my-sm-0" type="submit">Ara</button>
24. </form>
25. </div>
26. </nav>
```

**Şekil 14.15.** Duyarlı Menü Çubuğu Örnek Kodları.

Şekil 14.15.'teki kod blokunu incelediğimiz zaman menü çubuğunun genel yapısının 1. satırda tanımlandığını görüyoruz. Daha sonra sırasıyla çubuk içinde yer alacak elemanların tanımlamaları yer alıyor. 2. satırda “Logo” olarak kullanabileceğimiz bölüm, 3. ve 4. satırlarda ise görüntü alanı daraldığında menü çubuğunun nasıl bir davranış göstermesi gerektiği tanımlanıyor. Satır 7-20 arasında tanımlanan HTML liste öğeleri, bağlantı sağlanacak diğer sayfa ya da siteler için tıklanabilecek metinler oluşturuyor. Satır 21-24 arasında ise bir arama formu yer almaktır. Bu arama formu içerisinde bir metin kutusu ve tetikleme işlemini gerçekleştirecek bir buton yer alıyor.

Şimdi daha önceden hazırlamış olduğumuz duyarlı Web şablonumuza menü çubuğumuzu yerleştirelim. Sayfamızın görünümü Şekil 14.16.'daki gibi olmalıdır.



#### Bootstrap Nedir?

Bootstrap, mobil cihazlar için öncelikli ve duyarlı web siteleri geliştirmek için arayüz bileşen kütüphanelerine sahip bir çatıdır.

#### Bootstrap Paketi

Bootstrap paketinin içeriğini temel olarak iki farklı dosya türü oluşturur. Bu dosya türleri CSS tanımlamalarının yer aldığı \*.css uzantılı dosyalar ve

#### Bootstrap Kurulumu

Bootstrap kurulumu iki farklı şekilde gerçekleştirilebilir. Bunlardan ilki Bootstrap çekirdek dosyalarını İnternet üzerinde yer alan depolarдан

Şekil 14.16. Duyarlı Web Sayfasına Menü Çubuğu Eklenmiş Hâli.

Duyarlı Web sitemizin genel çatısını oluşturduk. Artık bu sayfanın *kopyalarını çıkartarak* ve çıkarttığımız kopyalara *menü çubuğundan bağlantılar* sağlayarak Web sitemizi tamamlayabiliriz. Kopyasını oluşturduğumuz sayfaların genel görünümlerinin de *aynı olmasına gerek yok*. Menü çubuğunu sabit bırakarak alt bölümde istediğimiz gibi sütunlar üzerinde oynayarak farklı bölmeler oluşturabiliriz. Şekil 14.17.'de Bootstrap'in 3. ve 4. Sürümlerini karşılaştırın bir Web sayfası örneği yer almaktır. Bu sayfada *slogan bölümü yok* ve sayfa 6 sütundan 2 bölmeye ayrılmış durumda.

BS

Ana Sayfa Bootstrap Hakkında Sürümler Yardım

Ara

#### Bootstrap Sürüm 3

- 4 katmanlı izgara (xs, sm, md, lg)
- Varsayılan yazı tipi boyutu 14px

#### Bootstrap Sürüm 4

- 5 katmanlı izgara (xs, sm, md, lg, xl)
- Varsayılan yazı tipi boyutu 16px

Şekil 14.17. Sayfanın İçeriğine Göre Farklı Tasarım Öğelerinin Kullanımı.

Bootstrap, gerek CSS tanımlamaları, gerekse de JavaScript kütüphaneleri ile çok hızlı bir şekilde duyarlı Web siteleri geliştirmemize olanak tanımaktır. Her yeni sürüm ile sahip olduğu özellikleri arttıran Bootstrap iyi bir kullanıcı topluluğu desteği de sahip. İnternet üzerindeki birçok kaynaktan Bootstrap ile ilgili ek özelliklere sahip *farklı çatılar* bulabilir ve bunları *mevcut yapıya entegre* de edebilirsiniz.

Bootstrap ile ilgili ücretli ya da ücretsiz bulabileceğiniz birçok tema da bu çatının limitlerini, yapılabilecek farklı çalışmaları ve görsel bileşenlerin detaylı kullanımını görebilmeniz açısından da faydalı olacaktır.



### Bireysel Etkinlik

- Proje klasörünüzde oluşturduğunuz "sablon.html" adlı bir dosyayı farklı kaydederek adını "index.html" olarak belirleyin.
- Sayfanızın gövde bölümünün en üstüne Şekil 14.15.'teki menü çubuğunu yerleştirin. Sitenizde yer alacak diğer sayfalara göre menü öğelerini yapılandırın.
- "index.html" adlı dosyadan kopyalar oluşturarak sitenizi tamamlayın.

## BOOTSTRAP WEB SİTESİ ÖRNEKLERİ

Bootstrap yaygın kullanımından dolayı İnternet üzerinde çok fazla örnek bulabileceğiniz bir çatı. Bu örnekler haricinde *beğendiğiniz herhangi bir Web sitesinin* kodlarına bakarak (ki bu kodları standart bir İnternet tarayıcıda görüntüleyebilmeniz için hiçbir engel yok), bu sitenin altyapısını hangi çatı ya da sistemlerin olduğunu da anlayabilirsiniz.

Bootstrap'in kendi Web sitesi içerisinde bulunan *örnekler* (examples) bölümü, yeni bir Web sitesinin tasarımına başlamak için ideal bir nokta. Bootstrap'in örnekler bölümüne <https://getbootstrap.com/> adresinde üst menüde yer alan "Examples" bağlantısı ile ulaşabilir ya da direkt olarak <https://getbootstrap.com/docs/4.3/examples/> adresini ziyaret edebilirsiniz (Şekil 14.18.).

The screenshot shows the Bootstrap Examples page at <https://getbootstrap.com/docs/4.3/examples/>. The page features a navigation bar with links to Home, Documentation, Examples, Themes, Expo, and Blog. Below the navigation, there are eight examples displayed in a grid:

- Album**: Simple one-page template for photo galleries, portfolios, and more.
- Pricing**: Example pricing page built with Cards and featuring a custom header and footer.
- Checkout**: Custom checkout form showing our form components and their validation features.
- Product**: Lean product-focused marketing page with extensive grid and image work.
- Cover**: A one-page template for building simple and beautiful home pages.
- Carousel**: Customize the navbar and carousel, then add some new content.
- Blog**: Magazine-like blog template with header, navigation, featured posts, and comments.
- Dashboard**: Basic admin dashboard shell with fixed sidebar and navbar.

**Şekil 14.18.** Bootstrap Örneklerinin Yer Aldığı Web Sayfası.

Şekil 14.18.'de görüntülenen sayfada yer alan örnekler, Bootstrap çatısına ek olarak *farklı CSS ve JS bileşenlerinin* de kullanıldığı örneklerdir. Bu açıdan bu örnekleri bir şablon dosya yapısı hâlinde kullanabilmek için *Web sayfasını tüm bileşenleri* ile birlikte bilgisayarınızın sabit diskine kaydetmek iyi bir başlangıçtır.



Bootstrap örnek Web sayfaları standart dosyalara ek olarak farklı bileşenler içerebilir.

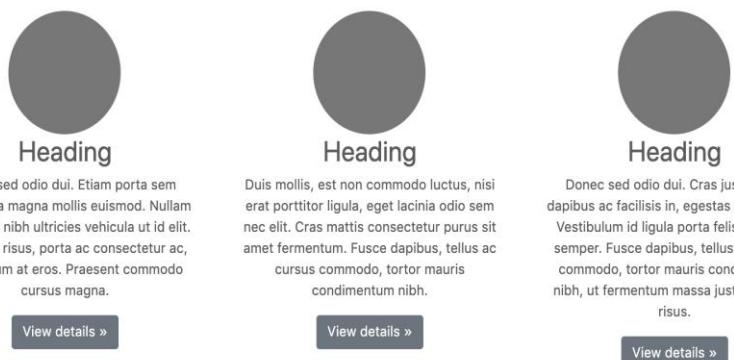
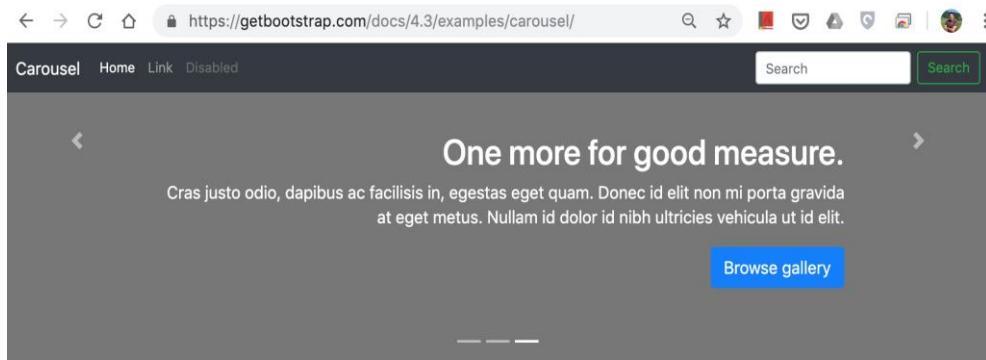


Bootstrap örneklerine <https://getbootstrap.com/docs/4.3/examples/> adresinden erişilebilir.

olacaktır. Bu işlem hemen her internet tarayıcısında benzer bir şekilde yapılmaktadır. “**Dosya > Sayfayı farklı kaydet**” menü öğelerini takip ederek örnek Web sayfası tüm bileşenleri ile kaydedilebilir.

Bootstrap örnekleri içinden günümüzde çok kullanılan bir temayı seçelim. Seçtiğimiz tema bir menü çubuğu, atlı karınca adı verilen (Carousel) bir slayt alanına ve içerik bölümünde sütunlu bir yapıya sahip olsun. Temanın genel görünümü Şekil 14.19.’daki gibidir ve aşağıdaki bağlantından erişilebilir:

<https://getbootstrap.com/docs/4.3/examples/carousel/>

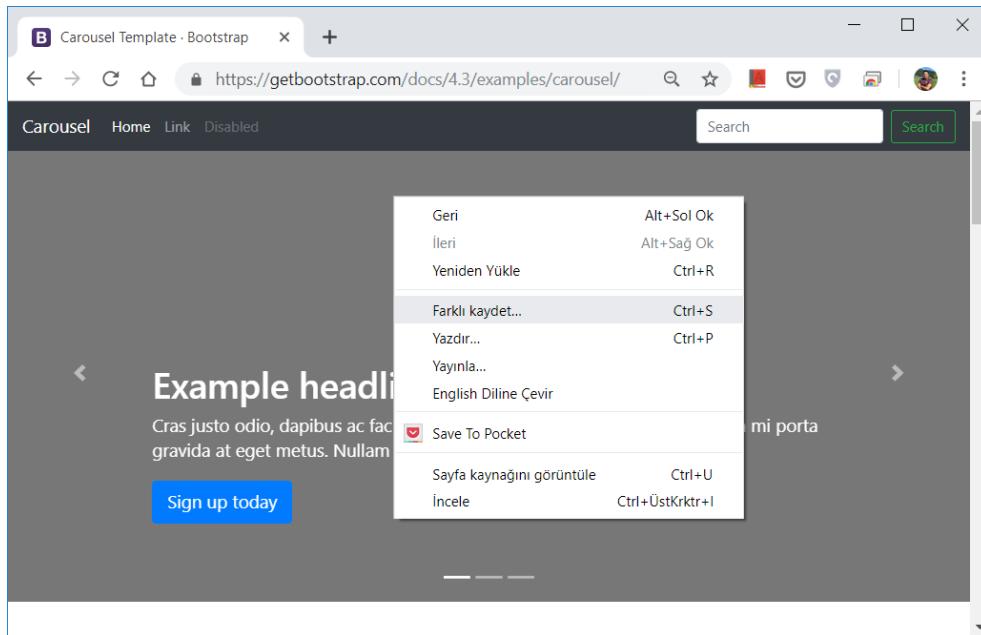


**Şekil 14.19.** Bootstrap Örnekleri, Atlı Karınca (Carousel) Teması.

Sayfayı internet tarayıcınızda görüntüledikten sonra sayfayı sabit diskinize kaydetmek için klavyeyinizin **CTRL + S** tuş kombinasyonunu kullanabilir (bu özellik **Google Chrome** ve **Mozilla Firefox** tarayıcılarında geçerlidir) ya da sayfanın boş bir noktasında farenize sağ tıklayarak açılan menüden “**Farklı kaydet**” menü öğesine tıklayabilirsiniz (Şekil 14.20.).

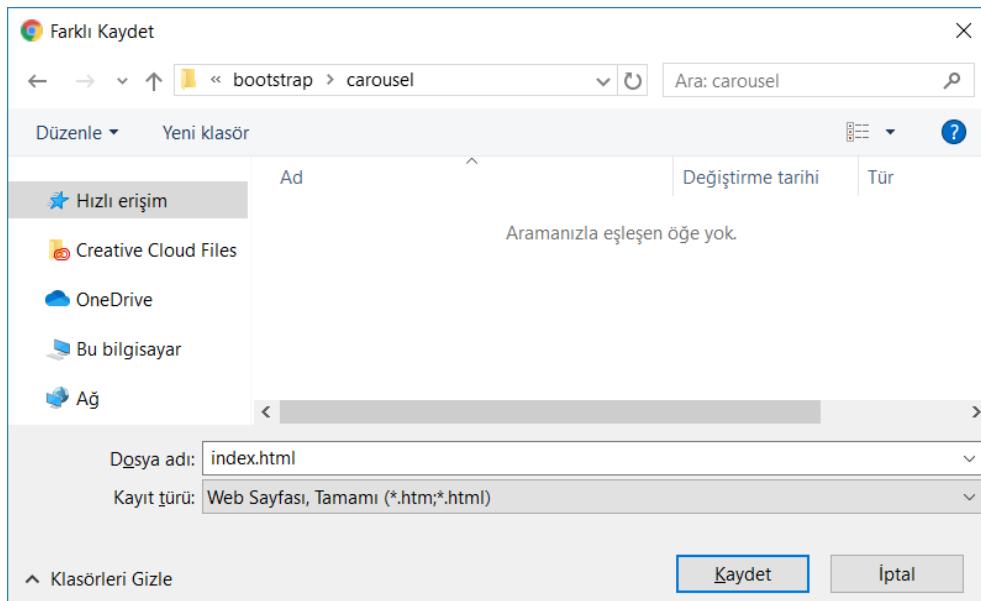


Tasarımını beğendiğiniz herhangi bir Web sayfasını internet tarayıcınız aracılığıyla sabit diskinize kaydedip inceleyebilirsiniz.



Şekil 14.20. Farenin Sağ Kлиgi İle Açılan “Farklı Kaydet” Menü Ögesi.

Sayfayı nereye kaydetmek istediğiniz soran bir *iletişim penceresi* açılacaktır. Bu pencere içerisinde sayfa ve bileşenlerini kaydetmek istediğiniz yeri seçiniz. Lokasyon olarak proje klasörünüz “*bootstrap*” içerisinde “*carousel*” adlı bir alt klasör oluşturabilir ve *tüm dosyaların* bunun içine kaydedilmesini sağlayabilirsiniz (Şekil 14.21.).



Şekil 14.21. Örnek Web Sayfasının Bilgisayar Sabit Diskine Kaydedilmesi.

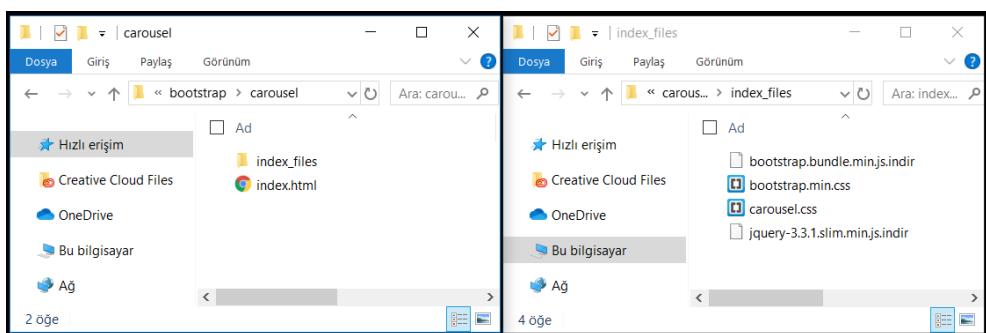
Şekil 14.21.’de dikkat etmemiz gereken iki önemli nokta var. Bunlardan ilki kaydederken Web sayfasına vereceğimiz ad. Bu ad genelde orijinal sayfanın başlık kısmında belirtilen ad olur. Bu adı örneğin “*index.html*” olarak değiştirebilirsiniz. Bu şekilde bileşenlerin kaydedileceği klasör adı Web için daha uyumlu bir adlandırma olur. Bu klasör adı daha sonra da *değiştirilebilir*, ancak o zaman sayfada verilen *referansları* da *güncellemek* gerekir.



Bir Web sayfاسını sadece HTML dosyası olarak ya da tüm bileşenleri ile (CSS, JS) bilgisayarınıza indirebilirsiniz.

İkinci nokta ise sayfa kaydedilirken iletişim penceresinin alt bölümünde yer alan biçim (format) kısmında “*Web sayfası, tamamı*” ibaresinin seçili olması. Bu ayar varsayılan olarak “Web sayfası, sadece HTML” şeklindedir. Bu şekilde bırakılırsa sadece HTML dosyası kaydedilir, sayfa içinde referans verilen bileşenler kaydedilmez. Bu durumda ihtiyacımız olan CSS ve JS dosyaları açısından İnternet üzerindeki kaynaklara bağımlı oluruz ve sitenin sonraki aşamalarında çeşitli sorunlarla karşılaşabiliriz.

Örnek Web sayfاسını tüm bileşenleri ile sabit diskimize kaydettiğimizde, elimizde bir HTML dosyası ve bileşenlerin yer aldığı bir klasör bulunmalıdır. “carousel” adlı yeni proje klasörümüzün ve bileşen alt klasörünün görünümü Şekil 14.22.’deki gibi olmalıdır.



Şekil 14.22. “Carousel” Projesi Klasör Yapısı Ve Dosyaları.

Şekil 24.22.’deki “index\_files” adlı klasörün içeriğini incelediğimizde hem Bootstrap ile ilgili dosyaları (*bootstrap.bundle.min.js*, *bootstrap.min.css*), hem jQuery kütüphanesini (*jquery-3.3.1.slim.min.js*), hem de bu projeye özel olarak kullanılan Carousel CSS dosyasını (*carousel.css*) görmekteyiz.



Bir Web sayfası içerisinde birçok farklı bileşen yer alabilir. Bu açıdan Web sayfalarını tüm öğeleri ile birlikte sabit diske kaydetmek önem taşımaktadır.

Örnek bir Web sayfاسını şablon olarak kaydetmek istediğimizde sayfayı tüm bileşenleri ile kaydetmenin önemini şimdi daha iyi anlamış olmalıyız. Sayfa içinde kullanılan onlarca farklı bileşen olabilir ve bu bileşenler farklı sürümleri ile proje içerisinde yer alabilir.

Örnek projemizin dosyalarını başarıyla kaydettiğimize göre artık proje klasörümüzde yer alan “*index.html*” adlı dosya üzerinde kendi istediğimiz değişiklikleri yapabiliriz. Menü çubuğundaki bağlantıları kendi Web sitemize göre düzenleyebiliriz. “index.html” dosyasını *şablon olarak* kullanıp diğer sayfalarımızı bu dosyayı temel olarak oluşturabiliriz.



## Özet

- Web sitelerine erişebilen cihazların farklı ekran boyutlarına sahip olduğu arayüz tasarımda birtakım sorunları da beraberinde getirdi.
- Önceleri masaüstü ve mobil cihazlar için ayrı ayrı yapılan arayüz tasarımlarını yerini Duyarlı Web Sitelerine bıraktı.
- Duyarlı Web Siteleri, ekran görünür alanına göre sayfadaki öğeleri yeniden konumlandıran bir yapının kullanılmasına olanak sağladı.
- Bu sayede farklı cihazlar için ayrı tasarım yapılması dönemi sona ermiş oldu.
- Bootstrap, mobil öncelikli olarak Duyarlı Web Siteleri geliştirmeyi kolaylaştıran bir HTML5 çatısıdır.
- Bootstrap JavaScript (JS) kütüphanelerinden ve CSS tanımlamalarından oluşur.
- Açık kaynak kodlu bir proje olan Bootstrap, birçok geliştirici tarafından tercih edilen popüler bir çatıdır.
- Bootstrap'in çalışma mantığı diğer HTML5 çatıları ile paralellik gösterir.
- Çalışmanın yapıldığı HTML dosyasından JS ve CSS dosyalarına referans verilerek bağlantı sağlama yeterlidir.
- Bootstrap çekirdek dosyaları, internet üzerindeki depolardan kullanılabileceği gibi, kişisel bilgisayarınızın sabit diskine kaydedilerek de kullanılabilir.
- Dosyalara referans verilmesiyle tamamlanan kurulum işleminden sonra ilk yapılması gereken Duyarlı Web Sayfası Şablonu'nun tasarlanmasıdır.
- Hazırlançık şablon daha sonra Web sitesinin iskeletini oluşturacak, farklı arayüz şablonlarına zemin teşkil edecektir.
- Gerek şablon, gerekse de şablondan üretilen dosyalar, internet tarayıcısı üzerinde test edilir.
- Bu test işlemi sırasında, internet tarayıcısının pencere boyutları değiştirilerek farklı cihaz ekranları simüle edilmiş olur.
- Bu işlem internet tarayıcının içeriğini yakınlaştırarak (zoom) da gerçekleştirilebilir.
- Yapılan çalışmaların farklı internet tarayıcılarında test edilmesi önemlidir.
- Mobil cihazlardaki internet tarayıcıları ile masaüstü sistemlerdeki internet tarayıcılarının davranışları kimi zaman farklılıklar içermektedir. Bu farklılıklar sonucunda yaşanabilecek sorunları en aza indirmek için test işlemleri ilgili cihazın kendi içinde de yapılabilir.
- Bootstrap, birçok farklı bileşenin yer aldığı örnek tasarımları da kendi web sitesinde bulundurur. Bu örnek çalışmalarla rahatlıkla erişilebilir ve bir şablon dosya olarak kullanılabilir.
- Örnek web sayfalarının şablon dosya olarak kullanılabilmesi için internet tarayıcılarının yardımıyla ilgili sayfanın kayıt edilmesi gereklidir.
- Kayıt işlemi sırasında dikkat edilmesi gereken nokta web sayfasının tüm bileşenleri ile kayıt edilmesidir.
- Kayıt işlemi gerçekleştirildikten sonra ana HTML dosyası üzerinde istenilen değişiklikler gerçekleştirilebilir ve kopyaları çıkartılarak web sitesinin diğer sayfaları bu şablon temel alınarak oluşturulabilir.

## DEĞERLENDİRME SORULARI

1. Aşağıdakilerden hangisi duyarlı bir Web sitesi tasarımının özelliklerini in içinde yer almaz?
  - a) HTML5 çatısına sahiptir.
  - b) Farklı Web yapılarına göre farklı bileşenler kullanılabilir.
  - c) Platformlar (İşletim sistemi, Internet tarayıcı vb.) arası kullanıma uygun bir yapıdadır.
  - d) Ekran boyutu farklı olan cihazlar için ayrı ayrı tasarıma ihtiyaç duyulur.
  - e) Görüntülendiği cihaza göre Web sayfasını yeniden yapılandırabilir.
2. Bootstrap kullanım hakları açısından aşağıdakilerden hangisi söylenebilir?
  - a) Açık kaynak kodludur.
  - b) Ticari amaçla kullanılamaz.
  - c) Kaynak kodların üzerinde değişiklik yapılamaz.
  - d) Kaynak kodlar kopyalanamaz.
  - e) Kaynak kodlar dağıtılamaz.
3. Bootstrap JavaScript ve CSS dosyalarının sıkıştırılmış sürümlerinin (min) sağladığı avantaj aşağıdakilerden hangisidir?
  - a) Boyut olarak daha küçük oldukları için Web sitesi optimizasyonu sağlar.
  - b) Dosyaların içeriğini herkes göremez, koruma sağlar.
  - c) Güncelleme işlemleri daha kolay gerçekleştirilir.
  - d) Farklı sunuculara aktarılması kolaydır.
  - e) Daha çok fonksiyonu içinde barındırabilir.
4. Bootstrap'i oluşturan çekirdek dosyaları İnternet üzerindeki depolardan kullanmanın avantajı aşağıdakilerden hangisidir?
  - a) Dosya boyutları daha küçük olur.
  - b) Birden çok kütüphane dosyası ile çalışmaya olanak sağlar.
  - c) Yedekleme işlemlerini hızlandırır.
  - d) Yazılan kodların daha hızlı çalışması sağlanır.
  - e) Dosyaların ilgili sürümlerinin her zaman en güncel hâlleri kullanılmış olur.
5. Bootstrap hangi bileşenlerden oluşur?
  - a) HTML, PHP, JavaScript
  - b) HTML, PHP, ASP
  - c) HTML, JavaScript, CSS
  - d) JavaScript, CSS, XLS
  - e) JavaScript, CSS, XML

6. Bir Bootstrap projesinde dosya tiplerine göre alt klasörler oluşturmanın avantajı nedir?
  - a) Web sitesinin toplam dosya boyutu küçülür.
  - b) Proje ilerledikçe artan dosya sayısının yol açacağı karmaşa önlenir.
  - c) Web sitesi kullanıcıları aradıklarını daha kolay bulur.
  - d) Web sitesinin güvenliğini arttırmır.
  - e) Web sitesinin daha hızlı açılmasını sağlar.
7. Duyarlı bir Web sitesi tasarımda menü çubuğu ile ilgili aşağıdakilerden hangisi söyledenemez?
  - a) İç içe menü öğelerinden oluşabilir.
  - b) Arama kutusu gibi farklı bileşenler içerebilir.
  - c) Ekran görüntülenebilir alanına göre menü çubuğu yeniden yapılanamaz.
  - d) Sabit bir genişliğe sahip olamaz.
  - e) Aynı sayfada birden fazla menü çubuğu kullanılabilir.
8. Bootstrap standart olarak kaç sütunlu bir yapı oluşturmaya olanak verir?
  - a) 3
  - b) 6
  - c) 9
  - d) 12
  - e) 16
9. Bootstrap ile oluşturulacak içerik alanında her biri 3 sütundan toplam kaç bölme oluşturulabilir?
  - a) 3
  - b) 4
  - c) 6
  - d) 8
  - e) 12
10. Aşağıdakilerden hangisi Bootstrap ile gerçekleştirilebilecek işlemlerden değildir?
  - a) Veri toplamak için formların oluşturulması
  - b) Verilerin yayınlanabilmesi için tabloların kullanımı
  - c) Site içi navigasyonun sağlanması
  - d) Prototip oluşturulması
  - e) Veri tabanı bağlantısı

Cevap Anahtarı

1.d, 2.a, 3.a, 4.e, 5.c, 6.b, 7.c, 8.d, 9.b 10 -

## YARARLANILAN KAYNAKLAR

- Bilge İş Web Tasarımının Temelleri Ders Notları. 18 Temmuz 2019 tarihinde  
<https://bilgeis.net/tr/courses/28/web-tasariminin-temelleri-html-ve-css>  
adresinden erişildi.
- Bootstrap Resmî Web Sitesi. 14 Temmuz 2019 tarihinde  
<https://getbootstrap.com/> adresinden erişildi.
- Bootstrap Sürüm 4 Dokümantasyon. 15 Temmuz 2019 tarihinde  
<https://getbootstrap.com/docs/4.3/getting-started/introduction/>  
adresinden erişildi.
- Bootstrap Web Sayfası Örnekleri. 17 Temmuz 2019 tarihinde  
<https://getbootstrap.com/docs/4.3/examples/> adresinden erişildi.
- Gelişken, U. (2016). Bootstrap 4: Responsive Tasarım Teknikleri ve Front-end Toolkit'leri. İstanbul: Level Yayınevi
- Mermerkaya, A. (2015). HTML5 & CSS3. İstanbul: Abaküs Kitap
- W3Schools Bootstrap Sürüm 4 Ders Notları. 14 Temmuz 2019 tarihinde  
<https://www.w3schools.com/bootstrap4/> adresinden erişildi.