

HTML

5

Simple Html 5

Book Version 1.0

Emre Can Öztas

Kapak Tasarımı: Samet KÖROĞLU

Simple HTML 5

Book Version: 1.0

Yazan: Emre Can ÖZTAŞ

Önsöz

Merhaba, ben Emre Can ÖZTAŞ. Bu kitabın yazarıyım. Gazi Üniversitesi – Bilgisayar Mühendisliği Bölümü'nde öğrenciyim. Bu kitap benim ikinci kitabım. İlk kitabım “Simple JavaScript” idi. Simple HTML 5 ile artık üzerimdeki o olumsuz etkiyi ve heyecanı atmış bulunmaktayım. İlk kitaptaki heyecanları yine yaşıyorum fakat artık acemi değilim.

Kitap yazmak, insanlara yardımcı olmak harika bir duygu. Geriye dönüp baktığımda iyi ki yazmışım diyorum. Bildiklerimi kitaplaştırma serüvenim sürüyor. Başka kitap projelerim de var.

Bu kitabı yazmamın 3 nedeni var. Bunlardan birincisi: Türkçe kaynağa destek olmak için, ikincisi: kitap alacak parası olmayan fakat öğrenmeye istekli arkadaşlarımızın faydalanması için, üçüncü neden: Allah nasip eder ilerleyen yaşlarımı görürsem geriye dönüp baktığımda neler yaptığımı hatırlamak için.

Biliyorum ki yazdığım bu kitaplar bazı insanlar tarafından eleştirilecek. Belki de yerden yere vurulup, eften püften sebeplerle suçlanacağım ama benim için önemli değil. Ben yoluma bakar ve işimi yaparım. Çünkü insanlar kendi yapamadıklarını, başkaları yaptığı zaman rahatsız olurlar. Ben hayatımı boş geçirmek istemiyorum. Hepimiz bir gün bu dünyadan gideceğiz. Ama ben arkamda iyi bir isim bırakarak ve belki de yardımcı olduğum insanlar tarafından hayır duasıyla anılacağım, inşAllah.

Simple HTML 5, yani bu kitabı takip edebilmek için temel HTML bilginizin olması gerekmektedir. Ayrıca bu kitap HTML değil HTML 5 üzerine kuruludur. Yani HTML 5 ile gelen yenilikleri anlatmak için yazılmıştır, bunu da belirtelim.

Bu kitap tamamen ücretsizdir. İçinde bulunan bölümlerin veya kitabın tamamının çıktısı alınabilir, kopyalanabilir, dağıtılabilir ve herkesle paylaşılabilir. Sizden istediğim yegane şey; bu kitabın tamamının veya bir bölümünün, herhangi bir yerde kullanılacaksa kaynak gösterilerek kullanılmasıdır.

Kitap hakkındaki görüş ve önerilerinizi, hatalı olan kodları veya yazım yanlışlarını lütfen bana bildirin. Mail adresim: emrecanoztas@outlook.com
Bir diğer konuda herhangi bir yerde takıldığınız zaman bana soru ile gelmenizi istemem. Ben sorularınıza içtenlikle cevap vermek, yardım etmek isterim fakat sizin takıldığınız yerde belki başka bir arkadaşımızda takılmış olabilir. O yüzden www.btsoru.com gibi platformda, bu takıldığınız sorunuzu sorarsanız, diğer arkadaşlarımıza da yardımcı olmuş olursunuz.

Teşekkürler

Bu kitabın yazımında bana olumlu / olumsuz destek olan çevremdeki herkese teşekkür ederim. Siz olmasaydınız; gelişimimi tamamlayamaz, bugünkü eriştiğim noktaya erişemezdim. İyi ki varsınız. Umarım hep var olursunuz.

Son olarak bu ve bir önceki kitabımın kapak tasarımını yapan Samet KÖROĞLU'na çok teşekkür ederim.

1	HTML	7
1.1	HTML Nedir?	7
1.2	HTML 5	7
2	HTML 5 için Tarayıcı Testi	9
3	HTML 5 için Araçlar	12
4	HTML 5 Sayfa Yapısı	15
4.1	DOCTYPE	15
4.2	lang	15
4.3	Character Encoding	17
4.4	<script>	17
4.5	<link>	18
4.6	HTML 5 Genel Sayfa Yapısı	18
5	New Elements	19
5.1	<article>	19
5.2	<aside>	20
5.3	<audio>	20
5.4	<canvas>	22
5.5	<command>	24
5.6	<datalist>	24
5.7	<details>	25
5.8	<embed>	27
5.9	<figure>	28
5.10	<figcaption>	30
5.11	<footer>	31
5.12	<header>	32
5.13	<hgroup>	33
5.14	<keygen>	34
5.15	<mark>	35
5.16	<meter>	36
5.17	<nav>	37
5.18	<output>	38
5.19	<progress>	39
5.20	<ruby>	40
5.21	<section>	41
5.22	<SVG>	42
5.23	<time>	43

5.24 <track>	44
5.25 <video>	45
5.26 <wbr>	46
6 New Form Fields & New Attributes	48
6.1 Form Fields	48
6.1.1 color	48
6.1.2 date	48
6.1.3 datetime	49
6.1.4 mail	49
6.1.5 list	50
6.1.6 number	51
6.1.7 month	51
6.1.8 URL	52
6.1.9 range	52
6.1.10 search	53
6.1.11 tel	53
6.1.12 time	54
6.1.13 week	54
6.2 New Attributes	55
6.2.1 autofocus	55
6.2.2 autocomplete	55
6.2.3 form	56
6.2.4 min & max	56
6.2.5 multiple	56
6.2.6 pattern	57
6.2.7 placeholder	57
6.2.8 required	57
6.2.9 step	57
7 Drag-N-Drop	59
8 Geolocation	63
9 Local Storage	67
9.1 Local Storage	67
9.2 Session Storage	71
10 Web Workers	73
11 New Events	78
12 HTTP Status Message	81

KAYNAKLAR	84
-----------	----

1.1 HTML Nedir?

HTML (Hyper Text Markup Language), Türkçe'ye çevirmek istersek; Yüksek Seviye Metin İşaretleme dili anlamına gelir. Web sayfalarını bir insana benzetirsek: HTML; bu insanın iskeleti, CSS; ete kemiğe bürünmüş hali ve JavaScript ise bu insanın hareket kabiliyetidir. Buradan çıkacak olan sonuç; HTML bir web sayfasının en temel yapısını oluşturur. Evet, HTML'yi bir iskelet gibi düşünebiliriz. Zira bir ev inşaa etmeye kalktığımızda ilk olarak o evin temelini hazırlarız ve iskeletini yaparız. Web sayfaları da aynen böyledir.

HTML bir programlama dili değildir. Bir Markup dilidir. Yani içeriğin nasıl ve nerde duracaklarını belirleme görevindedir. Bunun dışında herhangi bir görevi bulunmamaktadır. Sadece HTML ile hazırlanan sayfalar Statik durumdadır yani herhangi bir işlemi yerine getiremezler. Yalnızca nereye hangi içeriği eklemiş isek gövdesinde onları barındırırlar.

HTML, Berners Lee tarafından 1991 yılında geliştirilmiştir. Günümüze kadar farklı sürümleri çıkarılmıştır. Zira bu kitabın konusu da temel HTML değildir, HTML'nin son geldiği nokta olan HTML 5'tir.

Şimdi HTML'in sürümlerine bakalım.

HTML Version	Publish Date
HTML 1.0	1991
HTML 2.0	1995
HTML 3.0	1997
HTML 4.01	1999
HTML 5	2008

Yukarıdaki tabloyu incelediğimizde HTML'nin diğer sürümlerinin çıkış tarihlerini de görebiliyoruz. Burada demek istediğim başka bir şey var. Her yeni çıkan HTML sürümünde yenilikler gelmiştir fakat HTML bir bütündür. Bu zamana kadar çıkan Tags (Etiketler), hala kullanılmaktadır. Yani demek istediğim her yeni sürümde HTML'ye yeni etiketler ekleniyor. Bir anlamda; dönemin şartlarına göre yeni etiketler ekleniyor ve HTML genişletiliyor. Ve son olarak; HTML'nin gelişimi kesintisiz devam etmektedir.

1.2 HTML 5

HTML 5, 2008 yılında; W3C (World Wide Web Consortium) ve WHATWG (Web Hypertext Application Technology Working Group) tarafından geliştirilmiştir. Bu iki topluluk tarafından HTML 5'in geliştirilmesi kesintisiz bir şekilde devam edilmektedir.

HTML 5, bir çok yeniliği de beraberinde getirmiştir. Görünen o ki her yeni gelen versiyonla birlikte

HTML diğ er bir çok aracı da saf dış ı edecek. Örneğ in daha önce animasyon iç eren ya da baş ka bir deyiş le uç an-kaç an iç eriklerin olduğ u web sitelerinde sıklıkla Adobe F l a s h kullanılırdı. HTML 5 ile Flash ç öpe gitti diyebiliriz. Bunun dış ında Java'nın 1.2 sürümüy le ortaya çıkan App l e t'ler de artık tarihe karış mak üzere. Tabi ki bütün suç HTML 5'in değ il.

HTML 5 cıkalı çok olmadı fakat artık; hem yazılım geliřtiriciler hemde orta ve büyük çaplı platformlar web ortamlarına HTML 5'i dahil etmeye başladılar. Örneğ in F a c e b o o k artık HTML 5 kullanıyor. Bunu kullanıcı giriř ekranından bile anlayabilirsiniz. Bunun dış ında daha bir çok platform var.

HTML 5 ile birlikte JavaScript ve CSS artık üç ü ayrılmaz bir ikili oldu. HTML 5 ile JavaScript zaten bir yapbozu oluşturan parçalar gibi. Anlatmaya çalışacağ ım konularda da göreceksiniz bazı konularda JavaScript'ten destek alacağ ız.

Bunun dış ında HTML 5 adına söyleyecek çok söz var ama söyleceklerimi diğ er bölümlere bırakıyorum. Diğ er bölümlerde fazlaca değ ineceğ iz.

BÖLÜM 2:

HTML 5 için Tarayıcı Testi

Bu bölümde HTML 5 için tarayıcı testi yapacağız. Bu testi yapmamızın amacı; çoğu tarayıcılar HTML 5'e tam destek vermemektir. Bu test ile tarayıcıların HTML 5 için uygun olup olmadıklarını kontrol edeceğiz ve uygun olanlar arasında en yüksek orana ulaşan tarayıcı ile konularımızı işlemeye başlayacağız. Yani bu bir anlamda tarayıcı seçimi olacaktır.

Öncelikle HTML 5 tarayıcı testi için kullandığınız herhangi bir tarayıcı ile aşağıdaki adrese gidelim.

<http://html5test.com>

Hangi tarayıcı veya tarayıcıları test etmek istiyorsak, yukardaki vermiş olduğum adrese bu tarayıcılarla ayrı ayrı bağlanıp, bu tarayıcıların puanlarına bakmamız gerekiyor. Örneğin benim sistemimde; Chromium, Firefox Mozilla, Opera ve Midori tarayıcıları kurulu durumda. Ubuntu kullandığım için; Internet Explorer ve Yandex gibi tarayıcılar sistemimde bulunmamakta. Lakin sizin bilgisayarınızda kurulu ise bu tarayıcıları da test edebilirsiniz.

Yukarıdaki adrese bağlandığımızda tarayıcımızın, çeşitli HTML 5 elementlerinin hangilerine destek verip / vermediği gösterilmektedir. Bu sayfayı detaylı olarak inceleyebilirsiniz.

İlk olarak Firefox Mozilla ile başlayalım. [version: 39.0.3]

HTML5test - How well does your browser support HTML5? - Mozilla Firefox

YOUR BROWSER SCORES **466** OUT OF 555 POINTS

You are using Firefox 39.0 on Ubuntu

Save results Compare to... Share Donate

semantics

Parsing rules 5

<!DOCTYPE html> triggers standards mode	Yes	✓
HTML5 tokenizer	Yes	✓
HTML5 tree building	Yes	✓
HTML5 defines rules for embedding SVG and MathML inside a regular HTML document. The following tests only check if the browser is following the HTML5 parsing rules for inline SVG and MathML, not if the browser can actually understand and render it.		
Parsing inline SVG	Yes	✓
Parsing inline MathML	Yes	✓

multimedia

Video 29/35

video element	Yes	✓
Subtitles	Yes	✓
Audio track selection	No	✗
Video track selection	No	✗
Poster images	Yes	✓
Codec detection	Yes	✓
Advanced		
DRM support	Yes	✓

Yukarıdaki ekran alıntısında da görüldüğü gibi; Firefox Mozilla, 555 puan üzerinden 466 puanda. Yani genel olarak iyi bir puan fakat ne kadar yüksek olursa bizim için iyi olur. O yüzden diğer tarayıcılara da bakalım.

Opera'ya bakalım. [version: 12.16]

HTML5Test - How well does your browser support HTML5? - Opera

Opera | HTML5Test - How ... x |

Secure | html5test.com

YOUR BROWSER SCORES **338** OUT OF 555 POINTS

You are using Opera 12.16 on Linux

Save results | Compare to... | Share | Donate

semantics

Parsing rules 5

<!DOCTYPE html> triggers standards mode	Yes	✓
HTML5 tokenizer	Yes	✓
HTML5 tree building	Yes	✓
HTML5 defines rules for embedding SVG and MathML inside a regular HTML document. The following tests only check if the browser is following the HTML5 parsing rules for inline SVG and MathML, not if the browser can actually understand and render it.		
Parsing inline SVG	Yes	✓
Parsing inline MathML	Yes	✓

Elements 19/30

multimedia

Video 29/35

video element	Yes	✓
Subtitles	Yes	✓
Audio track selection	No	✗
Video track selection	No	✗
Poster images	Yes	✓
Codec detection	Yes	✓
Advanced		
DRM support	No	✗
Media Source extensions	No	✗
Codecs		
MPEG-4 ASP support	No	✗

Görüldüğü gibi Opera'nın puanı 338. Yani Firefox Mozilla'dan daha düşük.

Midori'ye bakalım. [version: 0.4.3]

HTML5Test - How well does your browser support HTML5?

Back | http://html5test.com/ |

YOUR BROWSER SCORES **351** OUT OF 555 POINTS

You are using Midori 0.4

Save results | Compare to... | Share | Donate

semantics

Parsing rules 5

<!DOCTYPE html> triggers standards mode	Yes	✓
HTML5 tokenizer	Yes	✓
HTML5 tree building	Yes	✓
HTML5 defines rules for embedding SVG and MathML inside a regular HTML document. The following tests only check if the browser is following the HTML5 parsing rules for inline SVG and MathML, not if the browser can actually understand and render it.		
Parsing inline SVG	Yes	✓
Parsing inline MathML	Yes	✓

Elements 24/30

multimedia

Video 33/35

video element	Yes	✓
Subtitles	Yes	✓
Audio track selection	Yes	✓
Video track selection	Yes	✓
Poster images	Yes	✓
Codec detection	Yes	✓
Advanced		
DRM support	No	✗
Media Source extensions	No	✗

Görüldüğü gibi Midori'nin puanı Opera'dan daha yüksek. Lakin pek çok insan adını bile duymamıştır, eminim.

Son olarak Chromium'a bakalım. [version: 43.0.2357.130]

HTML5test - How well does your browser support HTML5? - Chromium

HTML5test - How well does your browser support HTML5? - Chromium

https://html5test.com

YOUR BROWSER SCORES **501** OUT OF 555 POINTS

You are using Chromium 43.0.2357.130 on Ubuntu

Save results Compare to... Share Donate

JavaScript Diagrams

Click here for GoJS

semantics

Parsing rules 5

<!DOCTYPE html> triggers standards mode Yes ✓

HTML5 tokenizer Yes ✓

HTML5 tree building Yes ✓

HTML5 defines rules for embedding SVG and MathML inside a regular HTML document. The following tests only check if the browser is following the HTML5 parsing rules for inline SVG and MathML, not if the browser can actually understand and render it.

Parsing inline SVG Yes ✓

Parsing inline MathML Yes ✓

multimedia

Video 31/35

video element Yes ✓

Subtitles Yes ✓

Audio track selection No ✗

Video track selection No ✗

Poster images Yes ✓

Codec detection Yes ✓

Advanced

DRM support Yes ✓

Media Source extensions Yes ✓

Görüldüğü gibi yaptığımız testte en yüksek puanı Chromium aldı. Chromium, Google Chrome'nin GNU / Linux için uyarlamasıdır. Yani sisteminizde Chrome varsa HTML 5 için kullanabilirsiniz.

Yaptığımız testin sonuçları:

Sıra	Tarayıcı	Puan
1	Chromium	501
2	Firefox Mozilla	466
3	Midori	351
4	Opera	338

Bu kitabın konusu HTML 5 olduğu için biz de HTML 5 testinde en yüksek oyu alan Chromium ile konularımızı anlatmaya çalışacağız. Daha önce dediğimiz gibi sisteminizde Chrome varsa sizde Chrome ile derslerimizi takip edebilirsiniz.

BÖLÜM 3:

HTML 5 için Araçlar

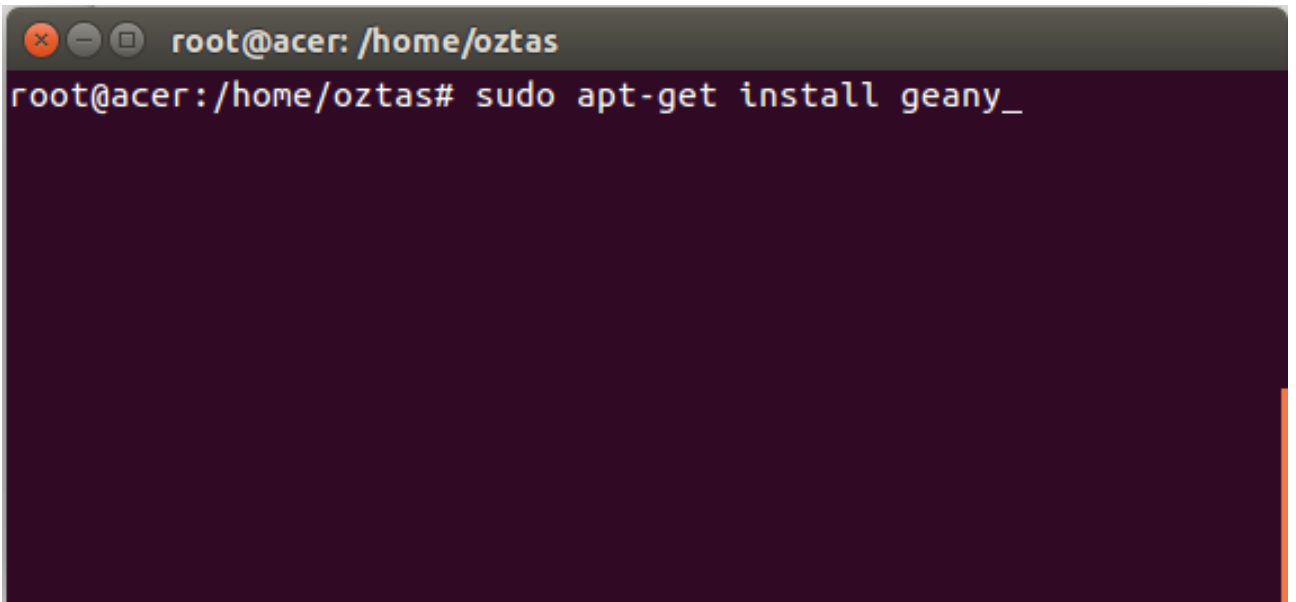
HTML programlamak için herhangi bir editor programını kullanabiliriz. Örneğin Windows kullanıcıları Notepad, GNU / Linux kullanıcıları da sistemlerinde öntanımlı gelen; Gedit, Kedit, Nano, Vim v.s gibi araçları kullanabilirler. Lakin her zaman iyi bir editor kullanmak akıllıca bir hareket olacaktır. Zaten Temel Web Programlama mantığını biliyorsanız yukarıdaki paragrafı yok sayabilirsiniz. Benim nazizane bir editor önerisinde bulunmak isterim: Geany.

Geany, hem GNU / Linux hemde Microsoft için sürümü bulunan ücretsiz bir programdır. Aynı zamanda birden fazla programlama diline hitap eden ve kod renklendirme özelliğine sahip bir text editor programıdır. Kod tamamlama özelliği maalesef bulunmamaktadır.

Peki neden Geany? Ben, tamamen bir Open Source (Açık Kaynak) destekçisi olduğum için herhangi ücretli bir araç kullanmıyorum. O yüzden güzide açık kaynak araçlarını kullanmaya özen gösteriyorum. Size de Geany'i tavsiye etmek isterim.

Microsoft kullanıcıları Geany'yi indirmek için aşağıdaki adrese girebilirler. <http://www.geany.org/>. Bunun dışında son bir öneri olarak Microsoft kullanıcıları, Geany'yi kullanmak istemezlerse Notepad++ programını da kullanabilirler.

GNU / Linux kullanıcıları ise Geany'i komut ekranından kurulumunu yapabilirler.

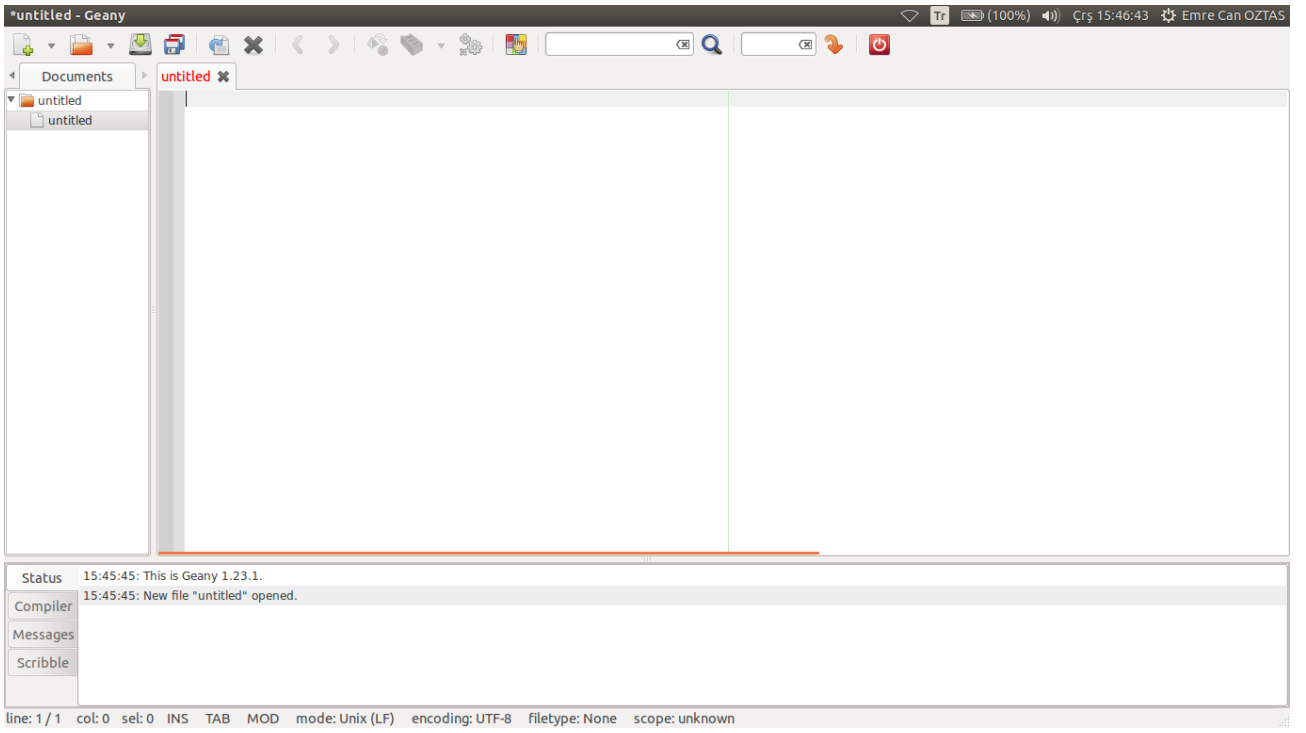


```
root@acer: /home/oztas
root@acer:/home/oztas# sudo apt-get install geany_
```



Yukardaki kurulum Debian tabanlı GNU / Linux dağıtımları içindir. Bu kitap baştan sona kadar Ubuntu kullanılarak hazırlanmıştır.

Geany kurulumundan sonra çift tıklayarak programımızı açalım.



Yukarıdaki ekran alıntısında gördüğünüz gibi Geany ortamı açıldı. Burada bir kaç ayar yapalım ve diğer bölümde JavaScript kodlamaya başlayalım.

Aşağıdaki ayarları daha rahat bir kod geliştirme ortamı için uygulayalım.

Öncelikli olarak aşağıdaki Message penceresini kapatalım:

Menü >> View > Show Message Window yolunu takip edip tik işareti kaldıralım.

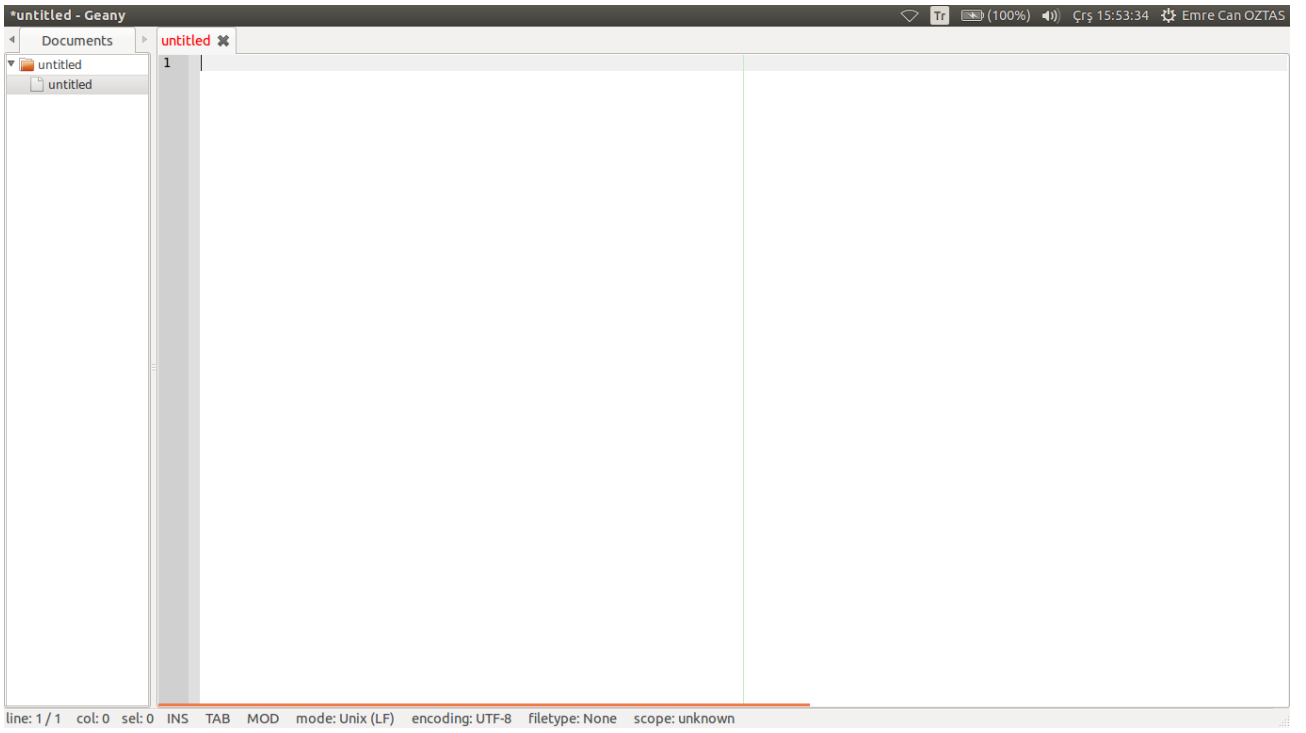
Araçlar çubuğunu kaldıralım:

Menü >> View > Show Toolbar yolunu takip edip tik işareti kaldıralım.

Satırları numaralandıralım:

Menü >> View > Editor > Show Line Number yolunu takip edip tik işareti koyalım.

Ayarlarımızı yaptık. Son olarak Geany aşağıdaki gibi görünecektir.



Geany ortamında Full Screen (Tam Ekran) çalışmak isterseniz F11 tuşunu kullanabilirsiniz.

Daha rahat bir geliştirme için Geany'i hazırlamış olduk. Son olarak HTML 5 ile çalışacağımızı Geany'e bildirelim. Bunun için aşağıdaki yolu takip etmemiz yeterlidir.

Menü >> Document > Set Filetype > Markup Languages > HTML source file

BÖLÜM 4:

HTML 5 Sayfa Yapısı

4.1 DOCTYPE

HTML 5 ile ilk yenilik DOCTYPE alanına gelen değişiklik olmuştur. Bilmeyenler için DOCTYPE'ı hemen açıklayalım. DOCTYPE, Browser (Tarayıcı)'a dokümanın tipini belirtir. Misal DOCTYPE'tan sonra yazacağımız ifade ile tarayıcımız bu dokümanın tipini anlar. Web sayfaları için HTML ifadesini kullanırız.



HTML, case-insensitive (Büyük / Küçük harfe duyarlı değil) idir. O yüzden etiketleri büyük / küçük harf farketmeksizin yazabilirsiniz.

Örneğin HTML 4.01'de DOCTYPE yazımı aşağıdaki gibiydi.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

Yukarıdaki yapımızda; URL, versiyon numarası, sayfanın dili v.s gibi bilgiler yer almaktadır. Çok uzun bir satır değil mi?

HTML 5 ile yukarıdaki gibi bir yapı kurmamıza gerek yoktur. Bunun yerine aşağıdaki gibi basit kullanım getirilmiştir.

```
<!DOCTYPE html>
```

Yukarıdaki yapımızda görüldüğü gibi sadece dokümanın tipinin html olduğu bildirilmiştir.

4.2 lang

HTML 5 ile sayfanın kodlama dili HTML etiketinde belirtilebilir. Örneğin HTML 4.01'de sayfa kodlaması !DOCTYPE satırında belirtiliyordu. Artık buna gerek olmadan aşağıdaki gibi sayfanın kodlama dili kolayca belirlenebilir.

```
<html lang="en">
```

Yukarıdaki örneğimizde sayfanın kodlaması olarak "en" yani "english - ingilizce" olarak belirledik. Eğer Türkçe olarak belirlemek istersek; "tr" yazmamız gerekir. Bu ve buna benzer diğer dil seçenekleri için aşağıdaki adresi ziyaret etmenizi tavsiye ederim.

Ülkelerin dillerinin, ISO – 639 – 1 altında kısaltılmıştır. Bu kısaltıların dillerden bazıları aşağıdaki gibidir.

Language	ISO Code
Afrikaans	af
Albanian	sq
Arabic	ar
Azerbaijani	az
Bulgarian	bg
Belarusian	be
Cambodian	km
Chinese	zh
Czech	cs
Danish	da
Dutch	nl
English	en
Finnish	fi
French	fr
Georgian	ka
German	de
Greek	el
Hindi	hi
Hungarian	hu
Icelandic	is
Indonesian	id, in
Irish	ga
Italian	it
Japanese	ja
Kannada	kn
Kazakh	kk
Kirghiz	ky
Korean	ko
Macedonian	mk
Malagasy	mg
Moldavian	mo
Mongolian	mn
Nepali	ne
Norwegian	no
Polish	pl
Portuguese	pt
Romanian	ro
Russian	ru

Serbian	sr
Slovak	sk
Slovenian	sl
Somali	so
Spanish	es
Swedish	sv
Tai	th
tibetan	bo
Turkish	tr
Turmen	tk
Uighur	ug
Ukrainian	uk
Uzbek	uz

4.3 Character Encoding

Character Encoding (Karakter Kodlaması), bir web sayfasının hangi dile destek vereceğinin belirtilmesidir. Örneğin sayfamızı Türkçe olarak tasarlamış isek ya da şöyle söyleyim; web sayfamızda Türkçe karakterler kullanmış isek buna uygun bir karakter seti kullanmalıyız. HTML 4.01'de bu işlemi META etiketinde belirtiyorduk. HTML 5'te yine META etiketini kullanıyorsunuz fakat HTML 4.01'deki gibi ayrıntılı değil.

Character Encoding: HTML 4.01

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

Yukarıdaki yapı ayrıntılı bir özellikteydi. Bu yapıyı hala da kullanabiliriz, kullanıyoruz. HTML 5 bu satırı biraz kısaltmıştır.

Character Encoding: HTML 5

```
<meta charset="UTF-8">
```

Yukarıdaki yapımız daha kısa ve daha kullanışlı bir yapıya sahip.

Hem HTML 4 hemde HTML 5 ifadesinde kullandığımız `charset` elementi ile sayfamızın dilini seçebiliyoruz. Bunu konumuzun başında belirtmiştik. UTF - 8 ifadesi bir Unicode karakter setidir. Bu karakter seti genel olarak önerilen karakter setidir. Bunun dışında da karakter setleri vardır. Bunlardan bazıları:

```
ISO-8859-1, US-ASCII, UTF-8, UTF-16, UTF-16BE - UTF-16LE
```

4.4 <script>

Bildiğiniz gibi `script` etiketi ile herhangi bir External (Harici) JavaScript dosyası çağrılabilir veya script etiketleri içerisinde doğrudan JavaScript kodlaması yapılabilir. HTML 5 ile script alanında bir takım değişiklikler olmuştur.

<script> etiketinin (HTML 4.01) kullanımı:

```
<script type="text/javascript" src="public.js"></script>
```

<script> etiketinin (HTML HTML 5) kullanımı:

```
<script src="public.js"></script>
```

HTML 5 ile type="text/javascript" özelliği atılmıştır. Yazılmasına gerek yoktur.

4.5 <link>

link etiketi ile External (Harici) CSS dosyaları çağrılabilir, bildiğiniz gibi. HTML 5 ile link alanında da bir takım değişiklikler olmuştur.

<link> etiketinin (HTML 4.01) kullanımı:

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

<link> etiketinin (HTML HTML 5) kullanımı:

```
<link rel="stylesheet" href="public.css">
```

HTML 5 ile type="text/javascript" özelliği atılmıştır, <script> etiketinde olduğu gibi. Yazılmasına gerek yoktur.

4.6 HTML 5 Genel Sayfa Yapısı

Bu bölümde verdiğimiz bilgiler ışığında HTML 5 ile kurduğumuz sayfa yapımızın örnek şablonuna bakalım.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Title here!</title>
<script src="public.js"></script>
<link rel="stylesheet" href="public.css">
</head>
<body>
...
</body>
</html>
```

Sayfamız yukarıdaki gibi bir şablona sahip olacaktır.

BÖLÜM 5:

New Elements

New Elements (Yeni Elementler veya Etiketler), bölümünde HTML 5 ile gelen yeni elementlerden bahsedeceğiz. Bu elementlerle birlikte HTML'nin yapabilecekleri artmış ve daha derli toplu bir yapıya kavuşmuştur.

HTML 5 ile gelen yeni elementleri sırasıyla anlatmaya başlamadan önce kitabın işlenişi hakkında bir açıklama da bulunalım. Elementlerimizi madde madde anlatmaya ve örneklerle desteklemeye çalışacağız. Her örneğimiz için ekran çıktısı almaya gerek görmüyorum. O yüzden önemli ve gerekli elementler için ekran çıktısı alacağız.

Şimdi hazırsanız; HTML 5 ile gelen yeni elementleri tanımaya başlayalım.

5.1 <article>

`article`, Türkçe karşılığı (makale) ile aynı görevdedir. Yani makale v.s gibi uzun yazıların bir arada bulunmasını sağlar. Örneğin herhangi bir blog sayfası veya forum sayfanız var ise bu elementi kullanarak daha modern bir yapıya kavuşabilirsiniz. Modern yapıdan kastım; `article` elementleri arasında yazılan ifadeler diğer elementler arasına yazılan ifadelerden, CSS ile daha farklı bir görünüme kavuşabilir ve tamamen bağımsızdır.

`<article>...</article>` elementlerinin kullanımı aşağıdaki gibidir.

```
<article>Burası bir makale alanıdır!</article>
```

`article` elementini kullanarak bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<article>
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non
proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
</article>

</body>
</html>
```

5.2 <aside>

`aside`, aynı içeriğe sahip ifadeleri bir araya toplamak için kullanılan bir elementtir. Böylelikle daha derli-toplu bir yapı oluşturulmuş olacaktır. Örneğin bir makalede belli içerikleri bir araya toplamak isteyebilirsiniz. Böyle durumlarda `aside` elementini kullanmak doğru bir seçim olacaktır.

`<aside>...</aside>` elementlerinin kullanımı aşağıdaki gibidir.

```
<aside>Burası bir toplanma alanı!</aside>
```

`aside` elementini kullanarak basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Untitled</title>
<meta charset="utf-8">
</head>
<body>

<article>This is Poem!<br>
<aside>
"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non
proident, sunt in culpa qui officia deserunt mollit anim id est laborum."
</aside>
</article>

</body>
</html>
```

Yukarıdaki örneğimizde `aside` elementini `article` elementi içerisinde kullandık. Bu ve buna benzer yapılar kurarak daha hoş ve fonksiyonel web sayfaları hazırlamamız mümkün.

5.3 <audio>

`audio` elementi ile belirtilen herhangi bir medya dosyası oynatılabilir. Örneğin; web sayfamızın dosyaları arasına konumlandığımız bir ses dosyasını `audio` elementi ile kullanıcının, kullanımına sunabiliriz.

`audio` elementi HTML 5'in en büyük yeniliklerinden birisidir. `audio` elementleri arasında eklenen medya içeriği tamamen kullanıcının kontrolündedir ve en önemlisi de bu medya dosyasını oynatmak için tarayıcıya herhangi bir `Plugin` (`Plugin`) kurmaya gerek yoktur.

`<audio>...</audio>` elementinin kullanımı aşağıdaki gibidir.

```
<audio>
<source src="medya_yolu" type="medya_tipi">
</audio>
```

Yukarıdaki yapımızı biraz inceleyelim. `audio` elementinin kapsam alanı içerisinde; `<source>` adında bir `child` (çocuk) elementi vardır. Buradaki `source` alanın `src` özelliğine hangi medya içeriğini eklemek istiyor isek; o dosyanın yolunu buraya eklememiz gerekir. `type` alanında ise o dosyanın

içeriğini bildirmeliyiz. Örneğin herhangi bir mp3 dosyasının yolunu src alanında belirtmiş isek, type alanına da audio/mp3 veya audio/mpeg yazabiliriz.

<audio>...</audio> elementinin kullanımı ile basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

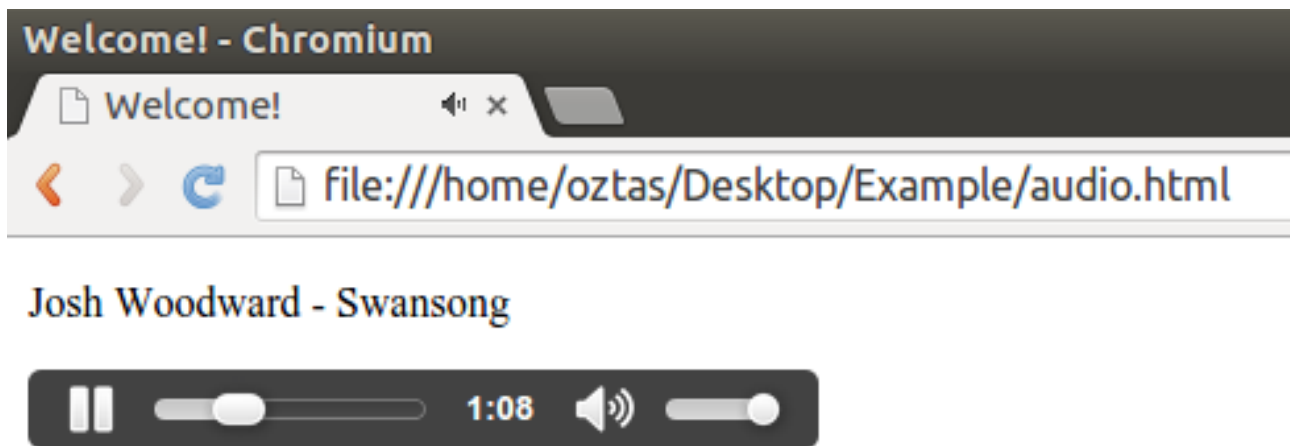
```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<p>Josh Woodward - Swansong</p>
<audio controls>
<source src="Josh Woodward - Swansong.ogg" type="audio/ogg">
</audio>

</body>
</html>
```

Yukarıdaki örneğimizde audio elementinin hemen yanına controls etiketini ekledik. Çünkü medya parçası çalınırken kontrol kullanıcı da olacak. control ifadesini silersek audio elementi bize yanıt vermeyecektir.

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



Yukarıdaki ekran çıktısında da görüldüğü gibi eklemiş olduğumuz medya içeriği çalınmaktadır. Yukarıdaki audio ile oluşturduğumuz çubuk size küçük gelmiş olabilir. CSS ile bu çubuğun width & height değerlerini kendinize göre ayarlayabilirsiniz. Örneğin ben aşağıda sizin için bir tane ayarladım.

```
<audio controls style="width: 500px; height: 20px;">
<source src="Josh Woodward - Swansong.ogg" type="audio/ogg">
</audio>
```



audio elementi ile sadece local'de değil herhangi bir web sitesindeki ses parçası da çalınabilir. Örneğin;

```
<audio controls autoplay>
<source src="http://thissite.com/audio1.mp3" type="audio/mpeg" />
<source src="http://thatsite.com/audio99.ogg" type="audio/ogg" />
</audio>
```

5.4 <canvas>

canvas, web sayfalarına basit şekiller çizmeye yarayan bir elementtir. Basitten kastım işte orta düzey. Örneğin bir kare, dikdörtgen v.s. Yani canvas'tan üst düzey bir performans beklememek lazım.

Bu element tek başına kullanılamaz. Bu yüzden bu element ile çalışmak için JavaScript'i kullanmamız gerekir. HTML 5 ile gelen bir başka çizim elementi de SVG'dir. SVG elementi ile canvas arasındaki farklardan bahsedeceğiz ve SVG elementini kullanmayı ilerleyen bölümlerde göreceğiz.

<canvas>...</canvas> elementinin standart kullanımı aşağıdaki gibidir.

```
<!-- canvas elementi olusturuldu -->
<canvas id="id_degeri" width="genislik" height="uzunluk"></canvas>

<!-- JavaScript ile canvas islemleri -->
<script type="text/javascript">

</script>
```



Canvas ile yalnızca 2 boyutlu şekil ve resimler üzerinde çalışılabilir.

Canvas elementini kullanarak basit şekiller çizdirmeye çalışalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<!-- canvas element -->
<canvas id="myShape" width="640" height="480"></canvas>

<!-- JavaScript ile canvas islemleri -->
<script type="text/javascript">

    /* ornek kullanim ve aciklama:

        # canvas elementinin degerini alalim
        > var context = document.getElementById("myShape");

        # canvas ile yalnızca 2 boyutlu sekiller cizdirilebilir.
        # 2 boyutlu sekiller üzerinde calisacagimizi bildirelim.
        > var ctx = context.getContext("2d");

        # olusturacagimiz sekli boyayalim
        ctx.fillStyle = "rgb(98, 150, 202)";

        # seklimizi olusturalim.
        ctx.fillRect(0, 0, 640, 480);

        internette basit bir arama ile canvas elementine ait
        bir cok ozellige ulasabiliriz.

        Dikkat! bazi tarayicilar 2 boyutu desteklemeyebilir.
        Bunun icin try...catch yapisi kullanmak yerinde olacaktır.

    */
```

```

var context = document.getElementById("myShape");

// evin dis yapisi
var disYapi = context.getContext("2d");
disYapi.fillStyle = "rgb(98, 150, 202)";
disYapi.fillRect(0, 0, 640, 480);

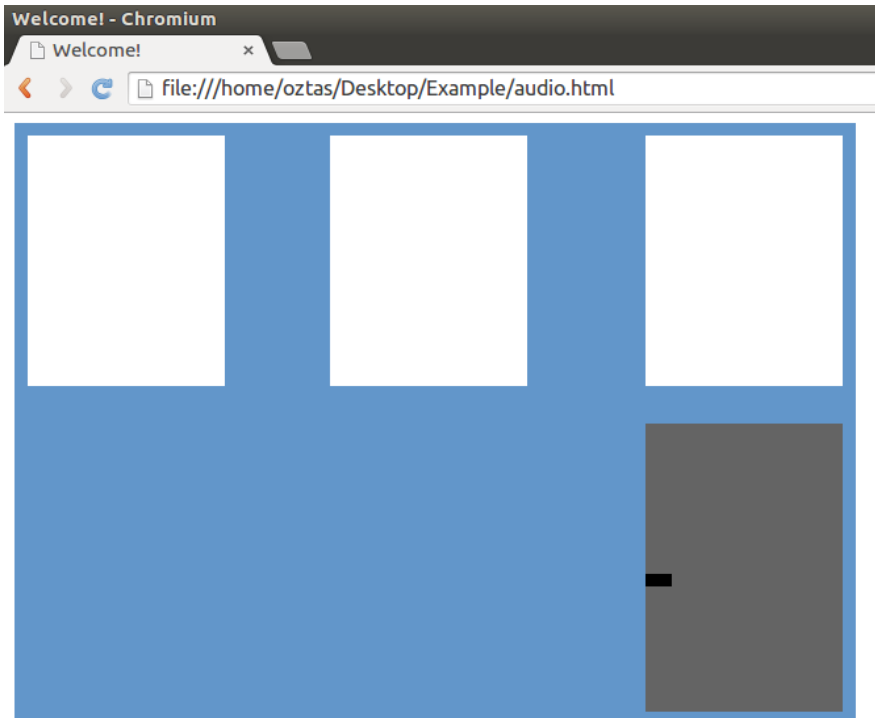
// pencereler
var pencere1 = context.getContext("2d");
pencere1.fillStyle = "rgb(255, 255, 255)";
pencere1.fillRect(10, 10, 150, 200);
var pencere2 = context.getContext("2d");
pencere2.fillStyle = "rgb(255, 255, 255)";
pencere2.fillRect(480, 10, 150, 200);
var pencere3 = context.getContext("2d");
pencere3.fillStyle = "rgb(255, 255, 255)";
pencere3.fillRect(240, 10, 150, 200);

// kapi
var kapi = context.getContext("2d");
kapi.fillStyle = "rgb(100, 100, 100)";
kapi.fillRect(480, 240, 150, 230);

// kapi kolu :))
var kapiKolu = context.getContext("2d");
kapiKolu.fillStyle = "rgb(0, 0, 0)";
kapiKolu.fillRect(480, 360, 20, 10);
</script>
</body>
</html>

```

Yukarıdaki örneğimizde bir ev çizdirmeye çalıştık. Bu örneğimize ait ekran çıktısını alalım.



Yukarıdaki ekran alıntısında da görüldüğü gibi evimizi canvas ve JavaScript yardımıyla çizdik. Bu çizim sadece prototip. Sizler, bizim canvas hakkında bahsetmediğimiz özelliklerini kullanarak değişik şekiller

elde edebilir ve hatta resim dosyaları üzerinde çalışabilirsiniz.

Aslına bakarsanız; canvas hakkında fazla bir şeyden bahsetmedik. Çünkü canvas geniş bir konu. Yani içine girdiğimiz zaman; bu kitabın amacından bize saptırarak kadar geniş. O yüzden canvas hakkında detay bir bilgi başka kaynaklardan almanızı tavsiye ederim.

5.5 <command>

command, elementi HTML 5'in komut elementidir. Örneğin herhangi bir dosya açmak isterken, resim yüklemek isterken veya başka herhangi bir işlemleri command elementinin işlemidir. Bu element hala geliştirilmektedir. Dolayısıyla çoğu web browser tarafından desteklenmemektedir.

<command>...</command> elementinin kullanımı aşağıdaki gibidir.

```
<command type="checkbox" label="click" onclick="function()">Click</command>
```

command elementi ile basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<!--
    type olarak: command, checkbox ve radio özellikleri kullanılabilir.
    label olarak, bir etiket belirlenmeli.
    events (olaylar) kullanılarak çeşitli işlemler yaptırılabilir.
-->
<command type="checkbox" label="click" onclick=alert('hello')>Click!</command>

</body>
</html>
```

5.6 <datalist>

datalist elementi, temel olarak form elementlerinin daha önce belirlenen herhangi bir eleman ile doldurulması işlemini gerçekleştirir. Örneğin formumuzda herhangi bir il liste alanı olsun. İşte bu liste alanını datalist ile kurduğumuz zaman kullanıcı daha önce belirlemiş olduğumuz illeri seçebilecek, aynı zamanda başka bir il değeri de alana girebilecektir.

<datalist>...</datalist> elementinin kullanımı aşağıdaki gibidir.

```
<datalist>
<!-- datalist için elemanlar option elementi ile belirlenir -->
<option value="deger1">
<option value="deger2">
<option value="deger3">
...
</datalist>
```

datalist elementi ile basit bir örnek yapalım örneğimiz aşağıdaki gibi olacaktır.

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>
<form action="#" method="POST">
<table style="border: 1px; border-style: dotted">
<tr>
<td>

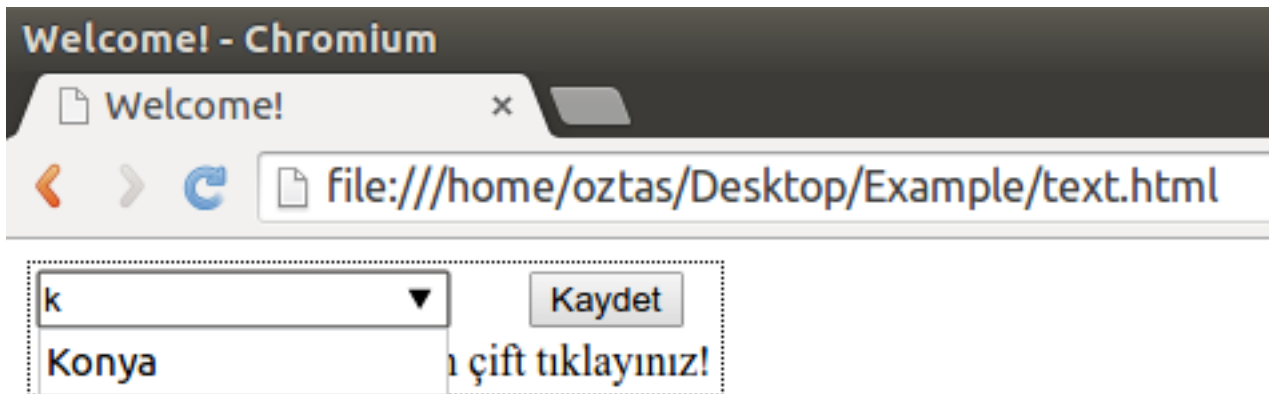
<!-- verileri tasimak icin bir list olusturalim. -->
<input list="sehirler">

<!-- verileri tutmak icin bir datalist olusturalim -->
<datalist id="sehirler">

<!-- verileri olusturalim -->
<option value="Ankara">
<option value="İstanbul">
<option value="İzmir">
<option value="Konya">
</datalist>
</td>
<td><input type="submit" value="Kaydet" /></td>
</tr>
<tr>
<td colspan="2">Formu tamamlamak için çift tıklayınız!</td>
</tr>
</table>
</form>
</body>
</html>

```

Yukarıdaki örneğimizin ekran çıktısını alalım.



Yukarıdaki ekran çıktısında görüldüğü gibi kullanıcı il değerini girerken otomatik olarak tamamlanıyor. Bir diğer konuda alanın hemen sağ tarafında bir ok işareti var. Bu işareti tıklayarakta daha önce datalist'te belirttiğimiz değerlerden herhangi birini seçebiliriz. Bazı tarayıcılarda bu ok işareti çıkmadığı için alana çift tıklamak gerekiyor. Zaten bunu da belirttik.

5.7 <details>

`details` elementi web sayfalarında belli alanlara açılır kapanır açıklama veya menüler eklemek için kullanılan bir yapıdır. Örneğin herhangi bir web sayfasında makalelerin ilk paragrafı gösterilir diğer paragrafları gizlenir. İşte böyle işlemler `details` elementi ile kolayca gerçekleştirilebilir.

`<details>...</details>` elementinin kullanımı aşağıdaki gibidir.

```
<details>
<!-- burasi bir
      acilir / kapanir
      alandir.
-->
</details>
```

details elementi yalnız başına kullanıldığında; açılır / kapanır alanın başlığı `De t a i l s` olur. Fakat summary elementi ile bu açılır / kapanır alana bir isim verilebilir. O yüzden summary elementi ile kullanmaya özen gösterelim.

summary elementi ile details elementinin kullanımı aşağıdaki gibidir.

```
<details>
<!--
      details icin, summary ile bir baslik ekleyelim.
-->
<summary>Burasi details icin bir baslikalani</summary>
</details>
```

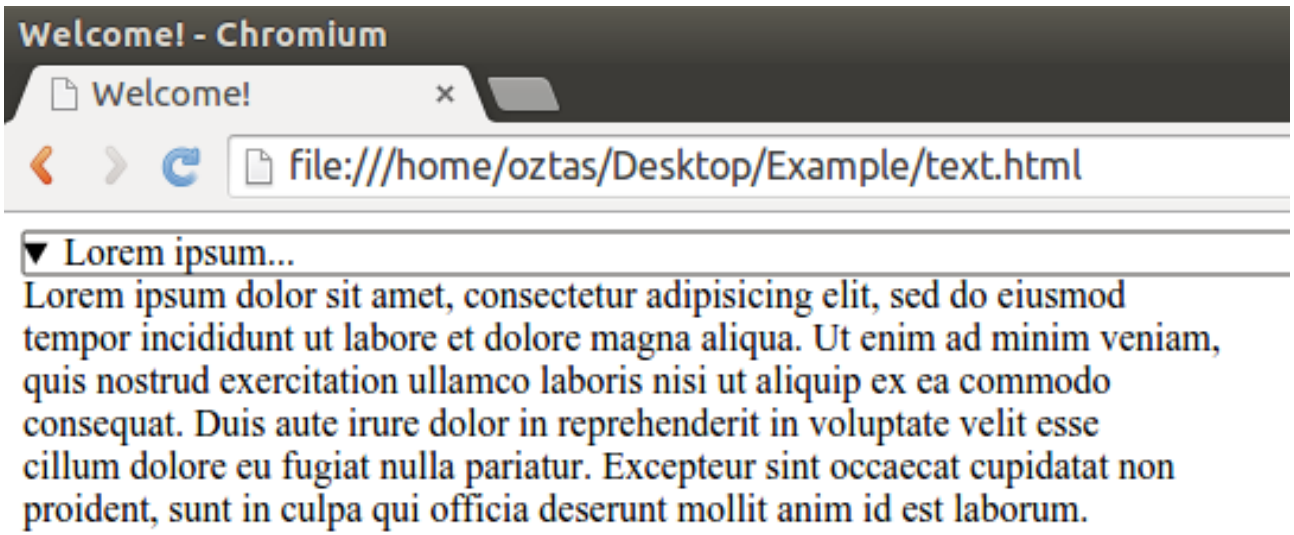
details elementi ile basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<details>
<summary>Lorem ipsum...</summary>
<article>
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod<br>
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,<br>
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo <br>
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse <br>
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non <br>
proident, sunt in culpa qui officia deserunt mollit anim id est laborum. <br>
</article>
</details>

</body>
</html>
```

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



Yukarıdaki ekran alıntısında da görüldüğü gibi açılır / kapanır menümüz hazır. Oldukça şık ve kullanışlı. HTML 5'ten önce bu tip açılır kapanır menüler JavaScript ile gerçekleştirilebiyorduk. Lakin şimdi JavaScript'e gerek kalmadan kolayca oluşturabiliyoruz.

5.8 <embed>

embed diğer bir medya çağırma / ekleme elementidir. Herhangi bir konumdaki medya dosyası çağrılabilir. Burada şunu da hatırlatmakta fayda var; embed ile eklenen medya içeriğine göre tarayıcılar tarafından ekstra plugin istenme durumları olabilir, baştan belirtelim. Zaten embed elementi de genellikle flash içerik eklemek için kullanılır.

<embed> . . . </embed> elementinin kullanımı aşağıdaki gibidir.

```
<embed src="medya_yolu"></embed>
```

embed elementinin bir özelliği olan src alanında eklemek istediğimiz medya dosyasının yolunu belirtmeliyiz.

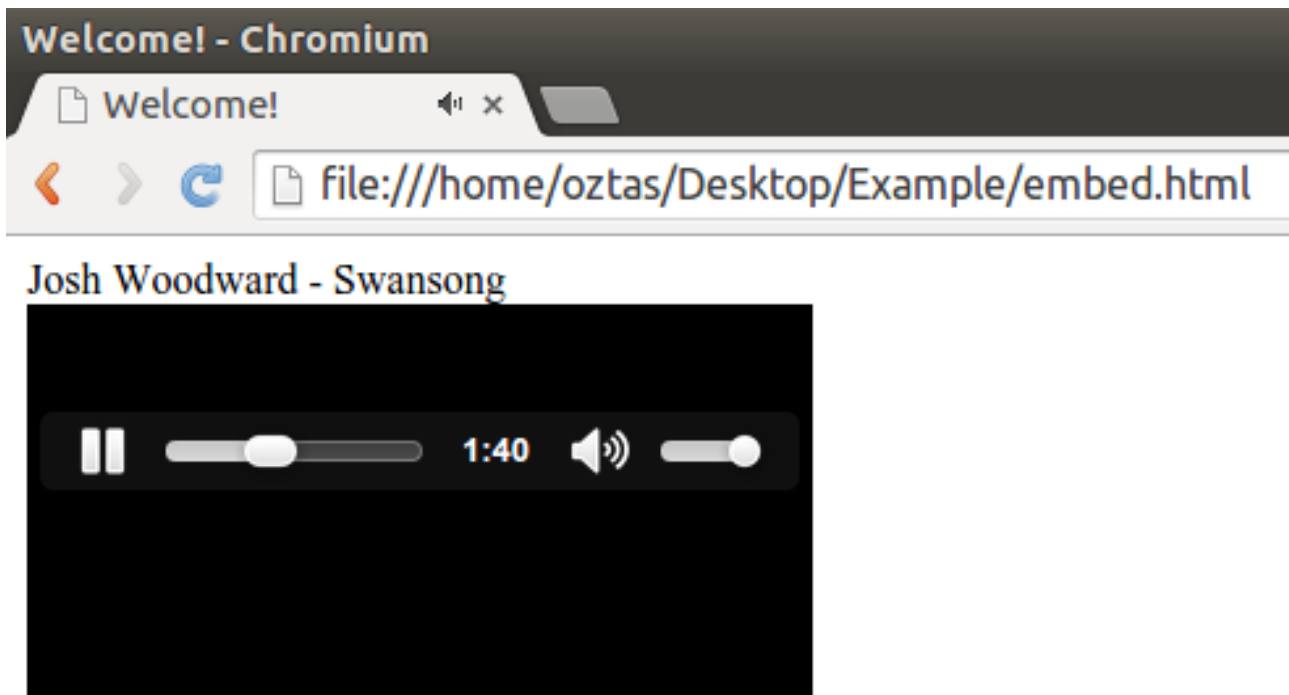
Örneğin <audio>...</audio> elementinde vermiş olduğumuz örneğimizi embed elementine göre revize edelim. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

Josh Woodward - Swansong <br>
<embed src="Josh Woodward - Swansong.ogg"></embed>

</body>
</html>
```

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



Yukarıdaki ekran çıktısını incelediğimizde; audio elementinden çok daha farklı bir şekilde olduğunu görüyoruz. Başta da belirttiğimiz gibi embed elementi flash içerikler eklemek için kullanılır.

5.9 <figure>

figure elementi, external (harici) ve internal (dahili) elemanlar için bir alan oluşturur. Bu oluşturulan alana container (taşıyıcı) denilir. Örneğin text ve resim dosyalarının birbirleriyle uyumu yakalamaları için figure elementi kullanılabilir.

<figure>...</figure> elementinin kullanımı aşağıdaki gibidir.

```
<figure>
<!--
    burasi external & internal
    elemanlar alanidir.
-->
</figure>
```

figure elementi ile basit bir örnek yapalım. Örneğimizi aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<!--
    figure elementinin kapsam alanı içerisinde;
    external & internal herhangi bir kaynak
    eklenebilir.
-->
<figure>
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod <br>
tempor incididunt ut labore et dolore magna aliqua.
```

```
Ut enim ad minim veniam, <br>
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo <br>
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse <br>
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non <br>
proident, sunt in culpa qui officia deserunt mollit anim id est laborum. <br>

<!--
    text'in altina herhangi bir resim
    dosyasini ekledik.
-->


</figure>

</body>
</html>
```

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



Yukarıdaki ekran çıktımızda görüldüğü üzere; figure elementine herhangi bir müdahalede bulunmadan kendi container'ını oluşturdu ve text & resim dosyasını kendine göre konumlandırdı. Gerçekten çok kullanışlı bir yapıya sahip.

5.10 <figcaption>

figure bölümünde bir container oluşturulup, bu oluşturulan alana external & internal olarak; resim, text, diğer elementleri kısaca içerik ekleyebiliyor idik. figcaption elementi ise bu figure elementi kapsamı içerisindeki elemanlara açıklama eklemek için kullanılan bir elementtir.

<figcaption>...</figcaption> elementinin kullanımı aşağıdaki gibidir.

```
<figcaption>Burasi bir aciklama alanidir!</figcaption>
```

figure elementindeki örneğimize figcaption elementini de ekleyerek düzenleyelim. Yeni örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<figure>

<!--
    figcaption ile text alanina bir
    aciklama ekledik.
-->

<figcaption>This is poem!</figcaption>

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod <br>
tempor incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, <br>
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo <br>
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse <br>
cillum dolore eu fugiat nulla pariatur.
Excepteur sint occaecat cupidatat non <br>
proident, sunt in culpa qui officia deserunt mollit anim id est laborum. <br>

<!--
    figcaption ile resme bir
    aciklama ekledik.
-->

<figcaption>He is Rocky!</figcaption>



</figure>

</body>
</html>
```

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



This is poem!

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
He is Rocky!



Yukarıdaki örneğimizde de görüldüğü üzere; figcaption ile yazdığımız açıklama satırlarımız belirttiğimiz alanlara konumlanmıştır.

5.11 <footer>

footer elementi, web sayfalarının alt kısmına içerik eklemek için kullanılan bir elementtir. Örneğin; footer ile lisans ve kullanım şartları, sayfanın amacı veya sayfayı hazırlayan kişinin bilgileri gibi çeşitli içerikler eklenebileceğini gibi herhangi bir medya içeriği veya resim de eklenebilir. Bu tamamen sayfayı tasarlayan kişiye kalmış bir durum.

<footer>...</footer> elementinin kullanımı aşağıdaki gibidir.

```
<footer>Burada sayfa sonu icerikleri var.</footer>
```

footer elementi ile basit bir örnek yapalım. Örneğimizi aşağıdaki gibi olacaktır.


```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<p>Lorem ipsum...</p>
<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod <br>
tempor incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, <br>
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo <br>
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse <br>
cillum dolore eu fugiat nulla pariatur.
Excepteur sint occaecat cupidatat non <br>
proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>

<!--
    footer elementini ekleyelim.
-->
<footer>Copyright &copy; Tüm hakları saklıdır.</footer>

</body>
</html>

```

Yukarıdaki örneğimizi incelediğimizde footer elementien alt satıra yazılmıştır. Çünkü bu elementin kapsam alanındaki içeriği sayfanın en alt satırında bulunacaktır. Tabiki bu elementi CSS kullanarak daha güzel bir görünüme kavuşturabiliriz.

5.12 <header>

footer elementi web sayfasının alt bant içeriklerini oluştururken; header elementi ise bu web sayfasının üst bandını oluşturur. Örneğin üst bantta; herhangi bir resim dosyası, medya içeriği, sayfanın başlığı veya menüler olabilir. İşte bu içerikleri header elementinde toplayabiliriz.

<footer>...</footer> elementinin kullanımı aşağıdaki gibidir.

```

<header>Burasi sayfa basi icerikleri var.</header>

```

header elementini kullanarak basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<!--
    header elementini ekleyelim.
-->
<header>Lorem ipsum...</header>

<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod <br>
tempor incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, <br>
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo <br>
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse <br>

```

```
cillum dolore eu fugiat nulla pariatur.
Excepteur sint occaecat cupidatat non <br>
proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>

<!--
    footer elementini ekleyelim.
-->
<footer>Copyright &copy; Tüm hakları saklıdır. </footer>

</body>
</html>
```

Yukarıdaki örneğimizde olduğu gibi; web sayfamızı hazırlarken, header ve footer elementlerini kullanmaya özen gösterelim. Böylelikle daha dertli bir yapıya kavuşmuş olacağız.

5.13 <hgroup>

Bildiğiniz gibi bir web sayfasında başlık oluşturmak için hazır olarak gelen <h> elementi var. Bu <h> elementi 1 – 6 arasında değer alır. Yani; <h1><h2><h3><h4><h5><h6>. İşte bu <h> elementlerini bir araya getirmek ya da başka bir deyişle gruplamak için hgroup elementi kullanılır.

<hgroup>...</hgroup> elementinin kullanımı aşağıdaki gibidir.

```
<hgroup>H >> 1 - 6 elementleri buraya gelecek.</hgroup>
```

Hgroup elementini kullanarak basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<!--
    h elementlerini gruplayalım
-->
<hgroup>
<h1>
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
</h1>
<h2>
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
</h2>
<h3>
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
</h3>
<h4>
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
</h4>
<h5>
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non
</h5>
<h6>
proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
</h6>
</hgroup>
</body>
</html>
```

5.14 <keygen>

keygen elementi, temel olarak key yani ardışık sıralamada birbirinden bağımsız anahtar sayılar üretmeye yarayan bir elementir. Keygen elementi 2 tip anahtar sayı üretir. Bunlar; `public` ve `private`. `public` key doğrudan server'a gönderilir. `private` key ise local key veri tabanında tutulurlar.

<keygen> elementinin kullanımı aşağıdaki gibidir.

```
<!--  
    keygen elementi form alanında  
    kullanılacağı için, bir name  
    vermemiz ve post/get ile gönderdiğimiz  
    sayfada yakalamamız gereklidir.  
-->  
<keygen name="keygen">
```

keygen elementini kullanarak basit bir örnek yapalım. keygen elementi, form alanında kullanılacağı için herhangi bir web programlama diliyle örneğimizi yapmamız gerekiyor. O yüzden ben bu örneğimizde PHP kullanacağım. Örneğimiz aşağıdaki gibi olacaktır.

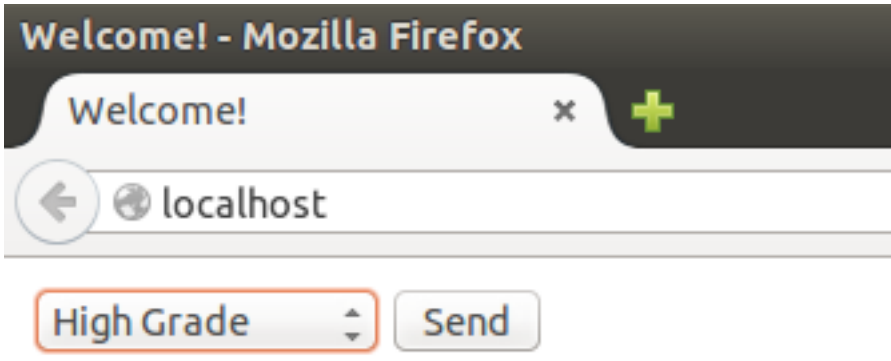
index.html;

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
<title>Welcome!</title>  
<meta charset="utf-8">  
</head>  
<body>  
  
    <!-- form alanı -->  
    <form action="keygen.php" method="POST">  
    <table>  
    <tr>  
    <!-- keygen elementini oluşturalım -->  
    <td><keygen name="keyPair"></td>  
    <td><input type="submit" value="Send"/></td>  
    </tr>  
    </table>  
    </form>  
  
</body>  
</html>
```

keygen.php:

```
<?php  
// index.html'den gelen keygen değerini alalım.  
$keyPair = $_POST["keyPair"];  
  
// aldığımız değeri ekrana yazdıralım.  
echo $keyPair;  
?>
```

Yukarıdaki örneğimize ait ekran çıktısını alalım.



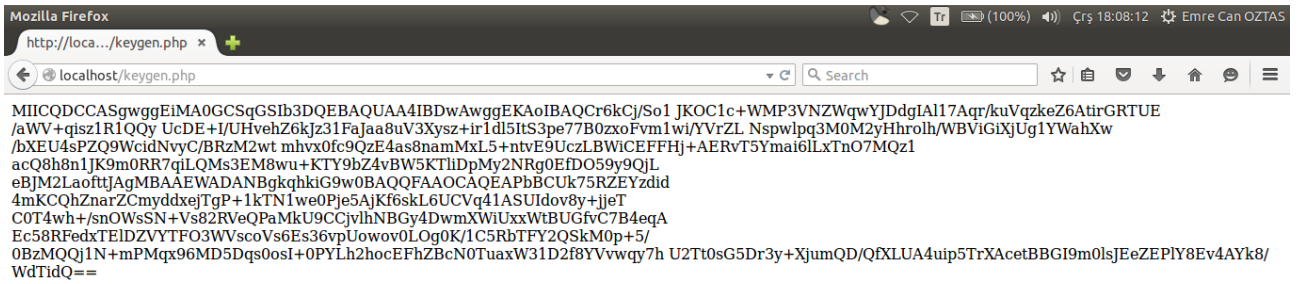
Yukarıdaki ekran çıktımızda; oluşturduğumuz keygen alanında iki seçenek var. Bunlar;

High Grade & Medium Grade

Bu iki seçenek arasındaki fark keygen ile üretilecek key'in uzunluğuyla alakalı. Ben High Grade seçeneğini seçiyorum ve Send butonuna tıklıyorum.

Keygen'in değer üretmesi, bilgisayarın performansına göre biraz zaman alıyor. Ayrıca bu örneğimizde Firefox Mozilla kullandım. Bunun sebebi Chromium'un sayıları formatlı göstermek yerine bir satır şeklinde göstermesidir, bunu da belirtelim.

Send butonuna tıkladıktan sonra oluşan key'imizin ekran çıktısını alalım.



Yukarıdaki ekran alıntısında da görüldüğü gibi keygen elementi, bize eşsiz bir key üretmiştir. Keygen elementini kullanarak çeşitli key'ler üretebilirsiniz.

5.15 <mark>

mark elementi temel olarak belirtilen alanın arkaplan rengi değiştirir ve vurgulama yapar. Mark elementinin çalışma prensibini fosforlu kalemlere benzetebiliriz. Hani ders çalışırken önemli satırların üzerine çizmek için kullandığımız o fosforlu kalemler. İşte mark elementi ile bir text içerisindeki önemli satırların arka plan rengini değiştirebiliriz.

<mark>...</mark> elementinin kullanımı aşağıdaki gibidir.

```
<mark>Buranin altini cizelim, onemli satir!..</mark>
```

mark elementi ile basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
```

```

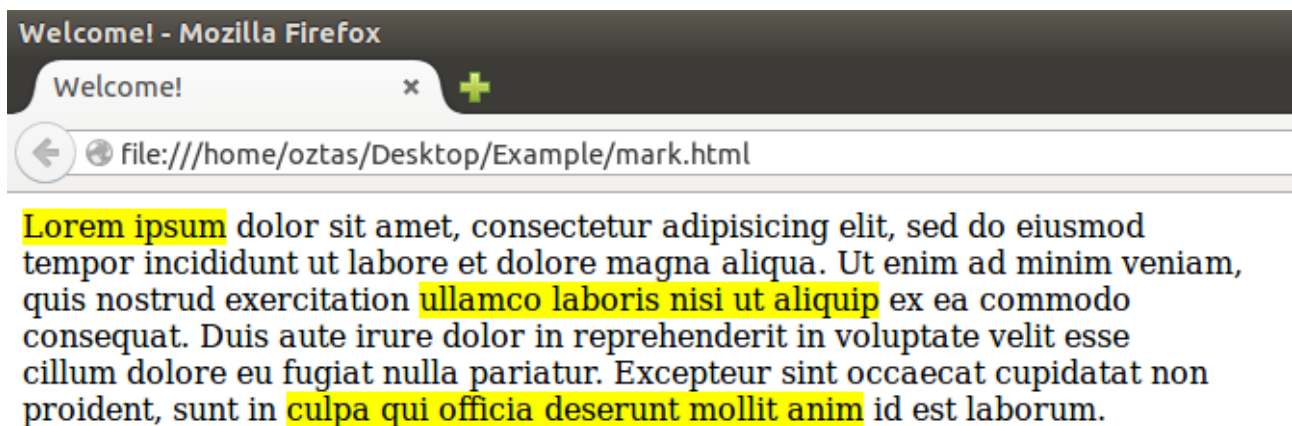
<meta charset="utf-8">
</head>
<body>

<article>
<mark>Lorem ipsum</mark> dolor sit amet, consectetur adipisicing elit,
sed do eiusmod <br>
tempor incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, <br>
quis nostrud exercitation <mark>ullamco laboris nisi ut aliquip</mark>
ex ea commodo <br>
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse <br>
cillum dolore eu fugiat nulla pariatur.
Excepteur sint occaecat cupidatat non <br>
proident, sunt in <mark>culpa qui officia deserunt mollit anim</mark> id est
laborum. <br>
</article>

</body>
</html>

```

Mark elementi ile yaptığımız örneğimizi ekran çıktısı aşağıdaki gibi olacaktır.



5.16 <meter>

meter elementi bir grafik elemanıdır. Örneğin herhangi bir değer için meter elementi kullanılarak bir görsel yapı oluşturulabilir. Ne demek istediğimi örneğimizi yaptıktan sonra daha iyi anlayacağınıza inanıyorum.

<meter>...</meter> elementinin kullanımı aşağıdaki gibidir.

```

<meter min="min_deger" max="max_deger" value="suanki_deger"></meter>

```

Yukarıdaki standart kullanım hakkında biraz konuşalım. min olarak belirtilen özelliğe görselin alacağı minimum değer, max olan belirtilen özelliğe maximum değer ve value özelliğine ise şuanki değer yazılmalıdır.

meter elemanı kullanarak basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">

```

```

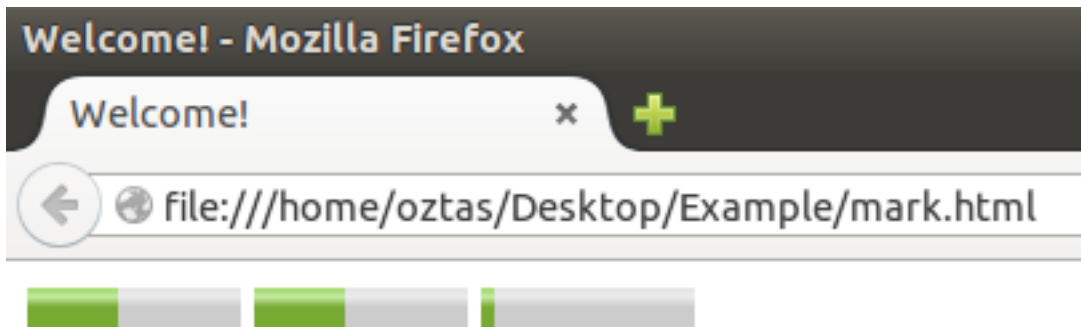
</head>
<body>

<meter min="0" max="100" value="42"></meter>
<meter min="0" max="10" value="4.2"></meter>
<meter min="0" max="100" value="6"></meter>

</body>
</html>

```

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



Yukarıdaki ekran alıntısında da görüldüğü gibi herhangi bir değer için meter elementiyle birlikte bir görsel çizdirebiliriz.

5.17 <nav>

nav yani navigator elementi, sayfadaki bağlantıları biraraya toplamak için kullanılan bir elementtir. Örneğin sayfasının başında veya sol / sağ tarafında oluşturulan linkler, nav elementi içerisine alınır, daha doğrusu alınmalıdır.

<nav>...</nav> elementinin kullanımı aşağıdaki gibidir.

```
<nav>Linkler burada yer alacak.</nav>
```

nav elementini kullanarak basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

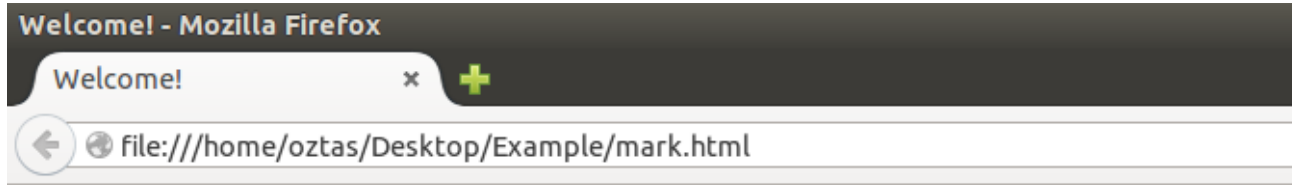
<header><h1>Emre Can OZTAS's Page!</h1></header>
<nav>
<a href="#">link1</a>
<a href="#">link2</a>
<a href="#">link3</a>
</nav>
<article>
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod <br>
tempor incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, <br>
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo <br>
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse <br>
cillum dolore eu fugiat nulla pariatur.

```

```
Excepteur sint occaecat cupidatat non <br>
proident, sunt in culpa qui officia deserunt mollit anim id est laborum. <br>
</article>
<footer>Copyright &copy; Emre Can OZTAS</footer>

</body>
</html>
```

Yukarıdaki örneğimizin ekran çıktısını alalım.



Emre Can OZTAS's Page!

[link1](#) [link2](#) [link3](#)

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
Copyright © Emre Can OZTAS

Yukarıdaki örneğimize baktığımızda; sayfamız fena olmamış diyorum. HTML 5 ile gelen elementleri etkin kullanarak daha iyi bir web sayfası tasarlamak artık çok daha kolay.

5.18 <output>

output elementi temel olarak web sayfasında gerçekleştirilen matematiksel işlemlerin sonuçlarını göstermek için oluşturulmuş bir elementtir. Bu element tek başına belirtilen matematiksel işlemleri gerçekleştirmiyor tabiki. output elementi sadece sayısal ifadelerin matematiksel sonuçlarının yer alacağı alanı temsil ediyor. Dolayısıyla işlemlerimizi JavaScript ile gerçekleştirip, output elementinin konumladığı alana sonucu yerleştirmemiz gerekli.

<output>...</output> elementinin kullanımı aşağıdaki gibidir.

```
<output>Burasi matematiksel sonucalani</output>
```

output elementi ile basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
<script type="text/javascript">
  var s1 = 0;
  var s2 = 0;
  var sonuc = 0;
  function hesapla(){
    s1 = parseFloat(document.getElementById("s1").value);
```

```

        s2 = parseFloat(document.getElementById("s2").value);
        sonuc = s1 * s2;
        document.getElementById("sonuc").innerHTML = sonuc;
    }
</script>
</head>
<body>

<table>
<tr>
<td><input type="number" value="0" placeholder="Birinci Sayı" id="s1"
onchange="hesapla()"/></td>
<td>&nbsp; * &nbsp;</td>
<td><input type="number" value="0" placeholder="İkinci Sayı" id="s2"
onchange="hesapla()"/></td>
<td>&nbsp; = &nbsp;</td>
<td><output id="sonuc"></output></td>
</tr>
</table>

</body>
</html>

```

5.19 <progress>

progress elementi, meter elementine benzer bir yapıda fakat; progress elementinde herhangi bir işin tamamlanmış kısmı v.s gibi durumlar gösterilir. Bir örnek vererek açıklamak istersek; herhangi bir program kurarken veya en basitinden oyun yüklerken karşımıza çıkan durum çubuğu HTML 5 ortamında progress elementi ile gelmiştir.

<progress>...</progress> elementinin kullanımı aşağıdaki gibidir.

```
<progress max="max_deger" value="simdi_deger"></progress>
```

progress elementinde min değeri bulunmaz. Çünkü her zaman 0'da başlar.

<progress>...</progress> elementini kullanarak çok basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```

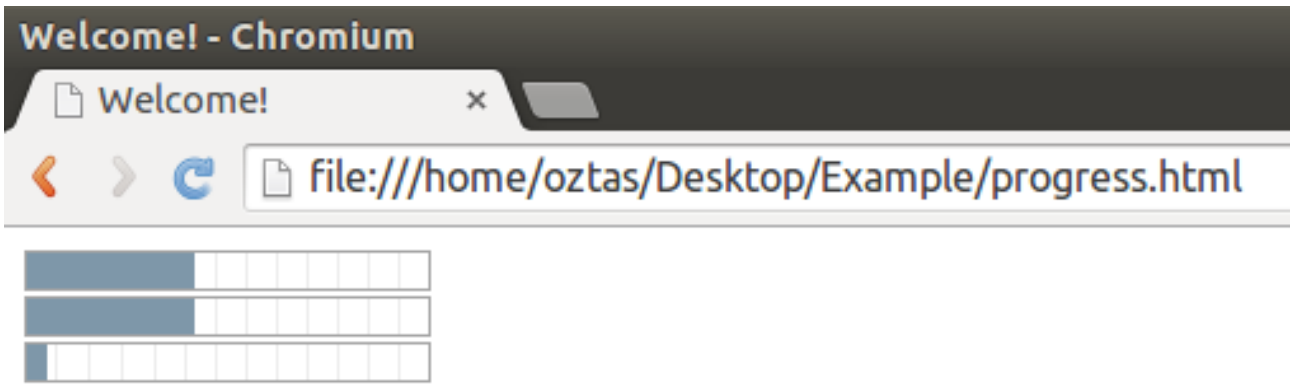
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<progress max="100" value="42"></progress> <br>
<progress max="10" value="4.2"></progress> <br>
<progress max="100" value="6"></progress> <br>

</body>
</html>

```

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



5.20 <ruby>

ruby elementinin, güzide programlama dili ruby ile herhangi bir alakası yoktur, baştan söyleyelim. ruby elementi temel olarak Doğu Asya Dilleri olan Japonca, Çince, Korece v.s için geliştirilmiş bir elementtir. Bildiğiniz gibi bu dillerde üstel işaretler oldukça fazladır. ruby elementinin sahip olduğu rt ve rb ile bu dillerdeki yazım sorunlarını önemli ölçüde çözer. Bunun dışında normal kullanımda da herhangi bir sıkıntı çıkarmazlar. Aksine daha göze hoş gelen bir yazım elde edilebilir. Ne demek istediğimi örneğimizden sonra daha rahat anlayacaksınız.

<ruby>...</ruby> elementinin kullanımı aşağıdaki gibidir.

```
<ruby>
<rb>taban ifadesi</rb>
<rt>üstel ifade</rt>
</ruby>
```

ruby elementinin sahip olduğu iki child(çocuk) elementi vardır. Bunlar <rb> ve <rt>. <rb> elementi, ruby base yani taban ifadesi olacaktır. <rt> elementi ise ruby text yani text ifadesi olacaktır.

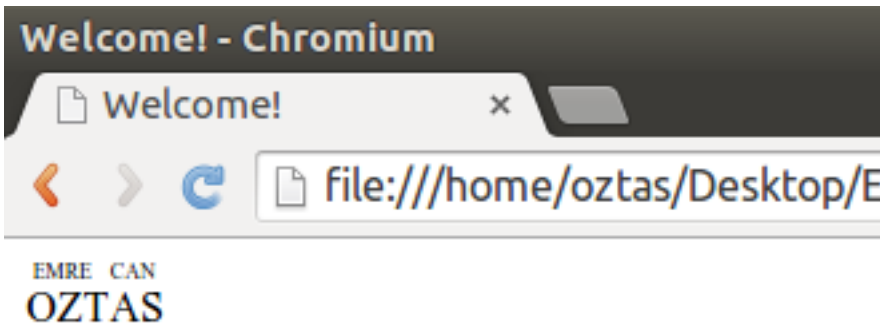
<ruby>...</ruby> elementi ile basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<ruby>
<rb>OZTAS</rb>
<rt>EMRE CAN</rt>
</ruby>

</body>
</html>
```

Yukarıdaki örneğimizin ekran çıktısını alalım.



5.21 <section>

section elementi, main (ana) yapıyı oluşturmak için kullanılan bir elementtir. Bildiğiniz gibi bir makale veya blog yazısını <article> elementi ile yazmalıyız. İşte article elementinin içerisinde section elementi ile ayrı ayrı alanlar oluşturabiliriz. section elementi <p> elementi ile karıştırılmamalı. Bu element sadece aynı tipteki içerikleri gruplamak için kullanılabilir.

<section>...</section> elementinin kullanımı aşağıdaki gibidir.

```
<section>Burasi bir icerik alanidir.</section>
```

section elementi ile basit bir örnek yaparak, kullanımını daha yakından inceleyelim. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<!-- article elementi ile bir makale baslatalim -->
<article>
<h1> Poem </h1>
<p> Hey there! this is for you! </p>
<section>
<h2>section: 1</h2>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod</p>
</section>
<section>
<h2>section: 2</h2>
<p>tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,</p>
</section>
<section>
<h2>section: 3</h2>
<p>quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo</p>
</section>
<section>
<h2>section: 4</h2>
<p>consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse</p>
</section>
<section>
<h2>section: 5</h2>
<p>cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat
non</p>
</section>
</section>
```

```

<h2>section: 6</h2>
<p>proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
</section>
</article>

</body>
</html>

```

5.22 <SVG>

SVG yani Scalable Vector Graphics elementi de canvas elementi gibi grafiksel işlemler yapmak için geliştirilmiş bir elementtir. canvas elementini JavaScript ile beraber kullanırken, SVG elementi tamamen XML tabanlıdır. Dolayısıyla JavaScript ile beraber kullanımı zorunlu değildir. Ayrıca SVG elementinin kullanımı hem daha basit hemde W3C organizasyonu tarafından tavsiye edilmektedir.

<SVG> . . . </SVG> elementinin kullanımı aşağıdaki gibidir.

```

<svg>...</svg>
<!-- svg'inin ozelliklerini belirtmeliyiz -->
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="250">

```

Yukarıdaki kullanım hakkında biraz konuşalım. İlk olarak; SVG, XML tabanlı bir element olduğu için xmlns özelliğinde svg ile çalışacağımızı bildirmemiz gerekir. Daha sonra version özelliğinde çalışacağımız svg versiyon numarasını girmeliyiz. Lakin bu zorunlu değildir. Tarayıcı hangi SVG versiyonunu tanıyorsa oSVG versiyonuyla da çalışabilir. Bir diğer alan olan height özelliğini ile çalışacağımız alanın uzunluğunu vermeliyiz. Genişliği belirtmeye gerek yoktur.

SVG elementi ile yapacağımız örneğimizi dikkatli takip edip anlamanızı tavsiye ederim. SVG ile oluşturabileceğimiz şekillerin bir bölümünü örneğimizde vereceğiz. Örneğimiz bol açıklamalı olacak. Örneğimizden bu kadar bahsettikten sonra aşağıdaki gibi bir örnek yazmış olacağız.

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<!--
    ilk olarak svg calisma alanlarimizi olusturalim.
    Not: fill ozelligi olusturulan seklin rengi verir.
    fill ozelligini kullanmadigimiz zaman siyah renk kendiliginden verilir.
    Ayrıca her sekil, kendine ait bir koordinasyona sahiptir.
-->

<!-- dikdortgen cizelim: rect -->
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="100">
<rect width="200" height="150" fill="gray" />
</svg>
<!-- daire cizelim: circle -->
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="200">
<circle cx="100" cy="100" r="100" fill="blue" />
</svg>
<!-- cizgi cizelim: line --><br>
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="100">
<line x1="10" y1="10" x2="300" y2="200" style="stroke: blue; stroke-width="10"/>
</svg>
<!-- polygon cizelim: polygon -->

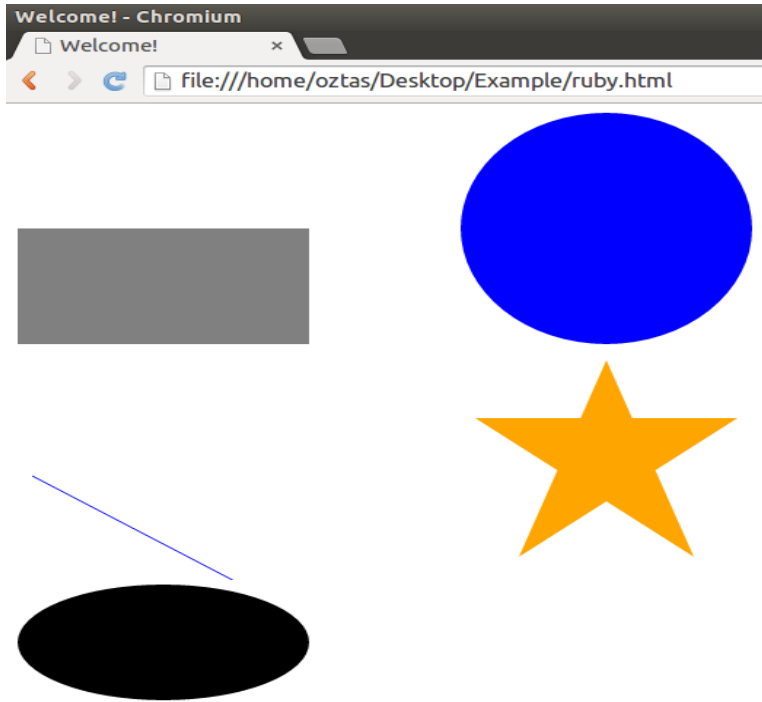
```



```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="200">
<polygon points="100,10 40,180 190,60 10,60 160,180" fill="orange"/>
</svg>
<!-- elips çizelim: ellipse --><br>
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="200">
<ellipse cx="100" cy="50" rx="100" ry="50" />
</svg>
</body>
</html>
```

Yukarıdaki örneğimizde; SVG elementi yardımıyla çeşitli şekiller çizdirdik. Bu şekillere yeni şekillerde eklenebilir. Lakin tam mantığını anlamaya çalıştığımız için bu kadar şekil bize yeterlidir.

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



5.23 <time>

`time` elementi, tarih ve saat yazımında kullanılan bir elementtir. Örneğin herhangi bir text içerisinde bir zaman belirtirken `time` elementi kullanılabilir. Hadd-i zatinde `time` elementinin sahip olduğu özellikler de vardır.

`<time>...</time>` elementinin kullanımı aşağıdaki gibidir.

```
<time>Tarih & Saat degeri</time>
```

`time` elementi ile basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>
```

```

<article>
<section>
<h1>What?</h1>
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non
proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
</section>
<section>
<!--
    time elementinin datetime özelliği vardır.
    fakat datetime özelliğinde yazılan değer web sayfasında gösterilmez
    bu yüzden kullanıcının bilgisayarının haberi olur.
-->
<time datetime="17.09.2015">17 Eylül</time>
</section>
</article>

</body>
</html>

```

5.24 <track>

track elementi; video veya ses dosyalarınıza alt yazı, açıklama, başlık, metada v.s gibi içerikler eklemek için kullanılan bir elementtir. Örneğin farklı bir dildeki videoyu web sayfanıza ekleyip, bu video için Türkçe dil paketleri hazırlayıp, bu dil paketini track elementi ile videonuza ekleyebilirsiniz. Zaten track elementi video ve ses dosyalarıyla birlikte çalışan bir elementtir.

track elementi henüz çoğu tarayıcı tarafından desteklenmemektedir.

<track>...</track> elementinin kullanımı aşağıdaki gibidir.

```

<track src="content_path" kind="content_type" srclang="en" label="English">

```

Yukarıdaki track elementinin kullanımını açıklamaya çalışalım. src özelliğinde ekleyeceğimiz içerik dosyasının yolunu vermeliyiz. kind alanında ise bu eklediğimiz dosyamızın tipini belirtmeliyiz. Örneğin bir alt yazı dosyası ekleyeceğimiz zaman kind özelliğine subtitles yazmamız gerekir. srclang alanında ise eklediğimiz dosyamızın dili ve label alanı da eklediğimiz içeriğin kısaca dilini belirtmek için kullanılan özelliklerdir.

track elementi ile bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<!-- video elementi -->
<video width="500" height="375" controls="controls">
<!-- icerik ekleyelim -->
<source src="3 Idiots.mp4" type="video/mp4" />
<!-- track elementi ile ekledigimiz icerige alt yazi ekleyelim -->
<track src="subtitles_en.vtt" kind="subtitles" srclang="tr" label="Turkish">
</video>

```

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

Yukarıdaki örneğimizde video elementini kullandık. Bunun yerine audio elementini de kullanabilirdik. video elementinden bir sonraki başlıkta detaylı olarak bahsedeceğiz.

5.25 <video>

video elementi, web sayfalarına video içeriği eklemek için kullanılan bir elementtir. Oldukça kullanışlı olan bu element ile herhangi bir dizindeki video eklenebileceği gibi herhangi bir web adresindeki video da eklenebilir. Örneğin YouTube.

<video>...</video> elementinin kullanımı aşağıdaki gibidir.

```
<video width="width_value" height="height_value" controls="controls"></video>
```

Yukarıdaki örnek kullanımda da görüldüğü gibi; eklenecek olan videomuzun, en & boy oranını ayarlayabiliyoruz. controls özelliğinden de kullanıcıya bu eklenen içeriğin kontrolünü bırakıyoruz.

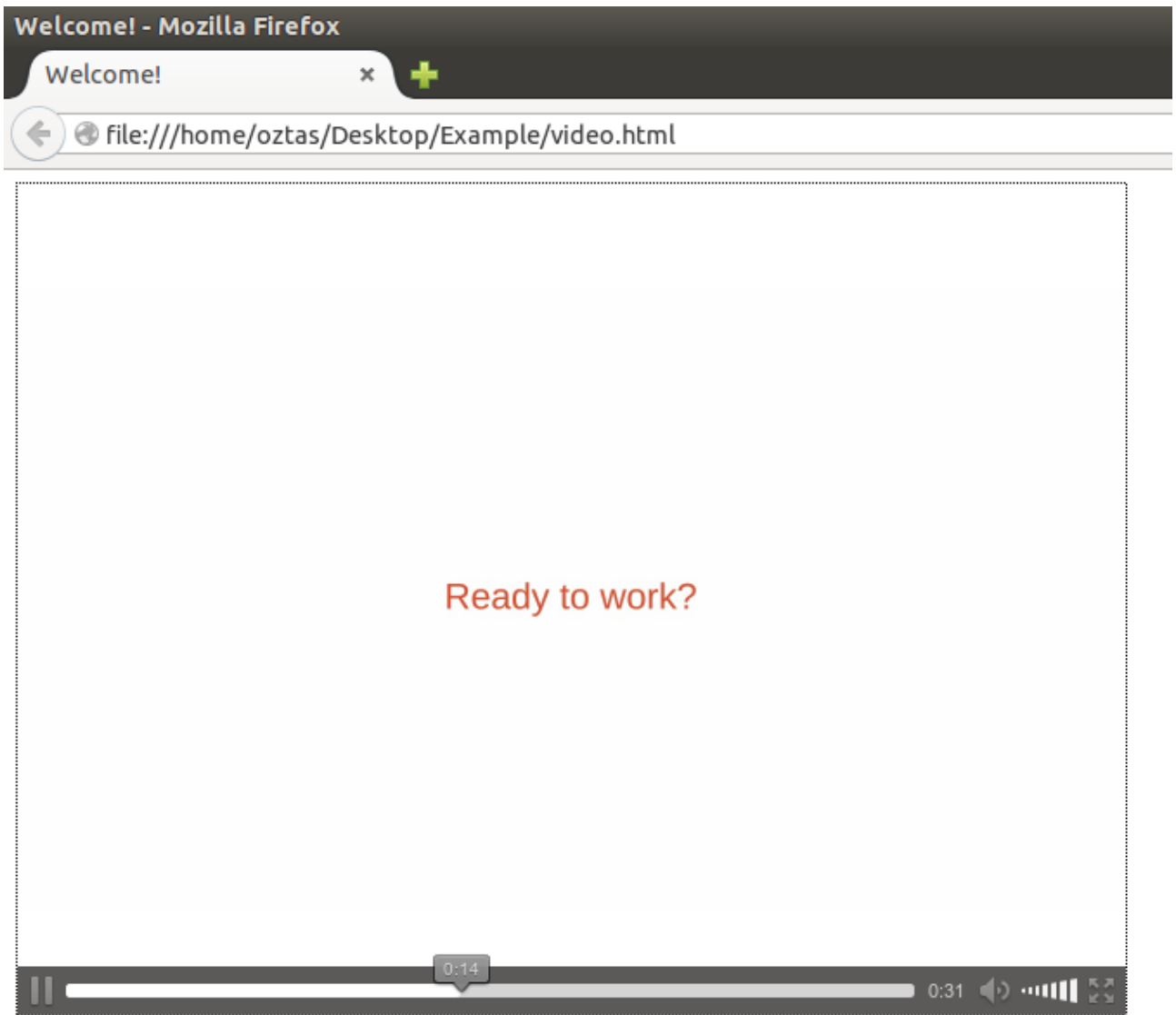
video elementi ile basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<!-- video elementini kullanarak video ekleyelim -->
<video width="640" height="480" controls="controls">
<!--
    source elementi icerik eklemek icin kullanilir.
    hem audio hem de video elementinde.
-->
<source src="How fast.ogg" type="video/mp4" />
</video>

</body>
</html>
```

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



5.26 <wbr>

İçeriklerimizi; <div>, <article>, <figure> v.s gibi elementlerin kap alanlarına yazarız. Çünkü böylelikle hangi içeriğin hangi işlemi yaptığı veya o içeriklerin şekillendirilmesi daha kolay olur. Bu ve buna benzer elementler container (taşıyıcı) elementlerdir. Bu elementlerin container sınırları CSS ile belirlenebilir. İşte bu noktada wbr elementi devreye girer. wbr elementi bu tip container'lardan taşan satırların bir alt satıra geçip içeriği yarıda kesmesini engelleyerek o satır boyunca yazılmasına devam eder. Bu element özellikle mail gibi önemli içerikler yazarken çok önemlidir.

<wbr>...</wbr> elementinin kullanımı aşağıdaki gibidir.

```
<wbr>Bu alan kesintiye ugramaz.</wbr>
```

wbr elementini kullanarak basit bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<body>

<header><h1>Emre Can OZTAS's page </h1></header>
```

```
<article>
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod <br>
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, <br>
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo <br>
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse <br>
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non <br>
proident, sunt in culpa qui officia deserunt mollit anim id est laborum. <br>
</article>
<footer>mail adresim: <wbr>emrecaoztas@outlook.com</wbr></footer>

</body>
</html>
```

BÖLÜM 6:

New Form Fields & New Attributes

Bu bölümde; HTML 5 ile gelen yeni Form Fields (Form Alanları) ve form alanlarının özelliklerini genişletmek için gelen Attributes (Özellikler) bahsedeceğiz. Aslında bu yenilikler HTML 4.01 üzerine kurulmuş bir yapıdır. Yani form alanında var olan özellikler korunup yeni özellikler eklenmiştir. Bu özelliklerin eklenmesiyle beraber HTML 5 daha özelleşmiş bir yapıya sahip olmuştur.



Bazı tarayıcıları, henüz bu yeni form alanlarını desteklemiyor. Eğer tarayıcının bu yeni form alanlarını desteklemiyorsa; bu form alanları normal bir text alanı gibi gözükecektir.

6.1 Form Fields

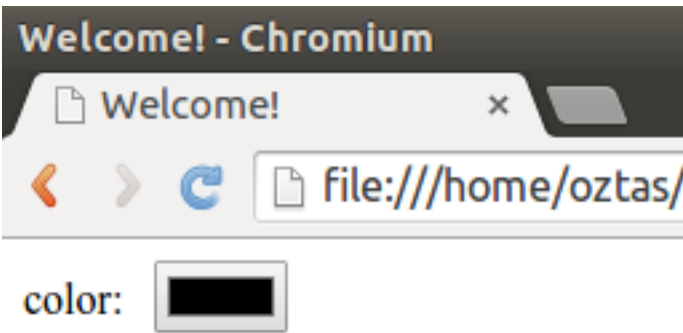
6.1.1 color

color, yeni input alanlarından bir tanesidir. Bu input alanı ile kullanıcı herhangi bir renk değerini seçebilir. Seçilen bu renk değeri sayfanın çeşitli yerlerinde kullanılabilir.

color form alanının kullanımı aşağıdaki gibidir.

```
<input type="color" name="color">
```

color input alanının görünüşü alttaki gibi olacaktır.



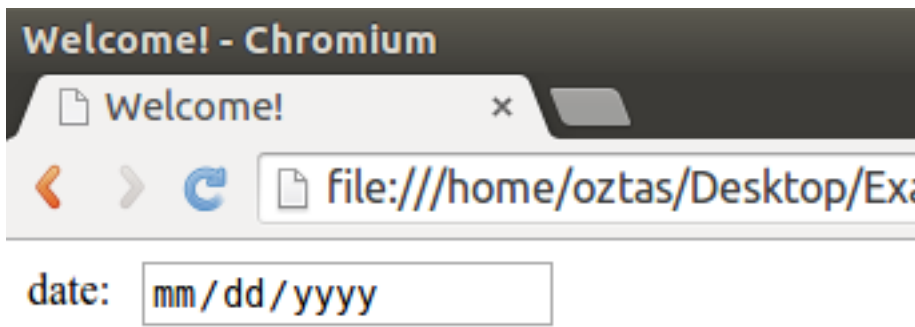
6.1.2 date

date, input alanı ile kullanıcıdan geçerli bir tarih değeri alınır.

date input alanının kullanımı aşağıdaki gibidir.

```
<input type="date" name="date">
```

date input alanının görünüşü alttaki gibi olacaktır.



6.1.3 datetime

`datetime` alanı, geçerli bir tarih ve saat değerleri girilmesi için uygulanan bir input alanıdır.

`datetime` input alanının kullanımı aşağıdaki gibidir.

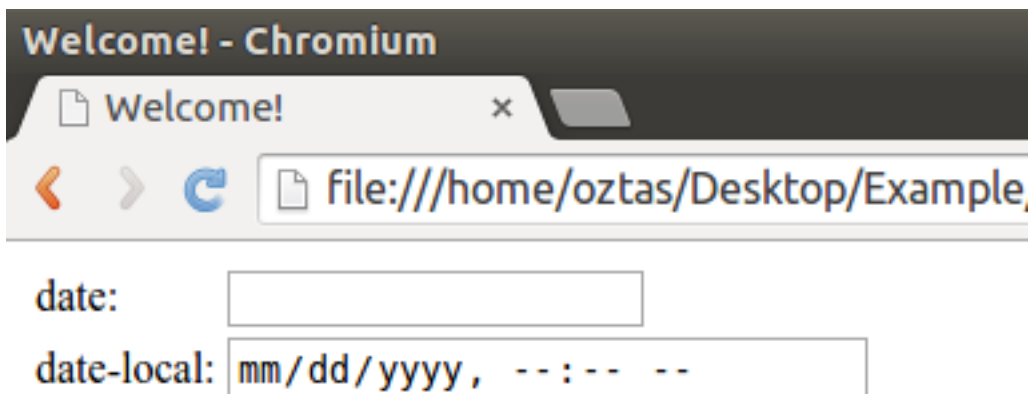
```
<input type="datetime" name="dateTime">
```

`datetime` alanından ziyade, doğrudan kullanıcıdan istenen tarih ve saat değerlerini almak için `datetime-local` alanı kullanılır.

`datetime-local` input alanının kullanımı aşağıdaki gibidir.

```
<input type="datetime-local" name="dateTimeLocal">
```

`datetime` ve `datetime-local` input alanlarının görünüşleri aşağıdaki gibi olacaktır.



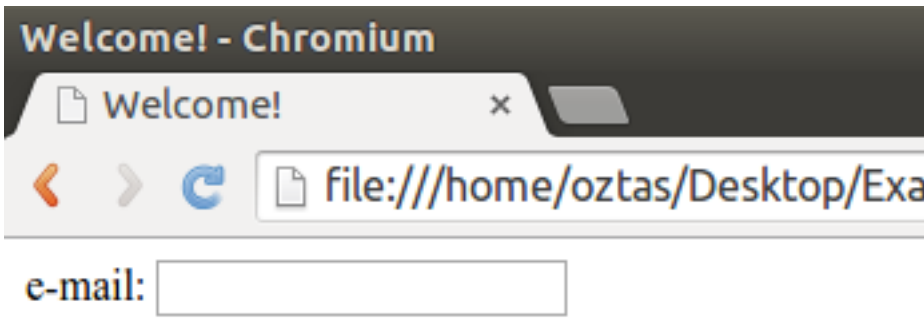
6.1.4 email

`email` input alanı kullanıcıların basit olarak mail adreslerini girmeleri için oluşturulmuş bir alandır. Basitten kastım, kullanılan tarayıcıdan daha önce herhangi bir mail adresi girilmiş ise bu mail adresini kullanıcı girerken gösterilir ve eğer istenirse tamamlanır.

`email` input alanının kullanımı aşağıdaki gibidir.

```
<input type="email" name="email">
```

`email` input alanının görünüşü aşağıdaki gibi olacaktır.



6.1.5 list

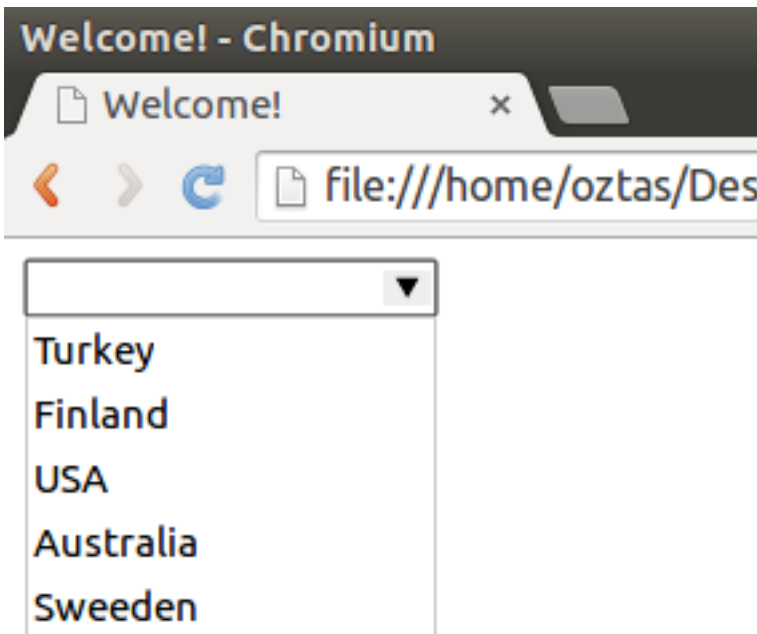
`list` alanı bir önceki bölümde değinmiştik lakin yine de değinelim. `list` alanı, aslında bir combobox gibidir. Yani daha önce tanımlanmış olan ifadeleri içerisinde barındırır ve kullanıcı gerekli gördüğü durumda bu ifadelerden herhangi birini seçebilir.

`list` input alanının kullanımı aşağıdaki gibidir.

```
<input list="listName">
<datalist id="ListName">
<option value="list1">
</datalist>
<!-- ya da -->
<input type="text" name="listField" list="comboBoxList">
<datalist id="comboBoxList">
<option value="">
</datalist>
```

Yukarıdaki kullanım stillerinden her ikisi de kullanılabilir. Çünkü `list` bir `type="text"` alanının özelliğidir. Burada dikkat edilmesi gereken nokta; `list` ve `datalist` `id` değerlerinin birbiriyle aynı olmasıdır. İki ifade farklı olursa tarayıcı bu iki ifadeyi birbiriyle eşleştiremez.

`list` input alanının görünüşü aşağıdaki gibi olacaktır.



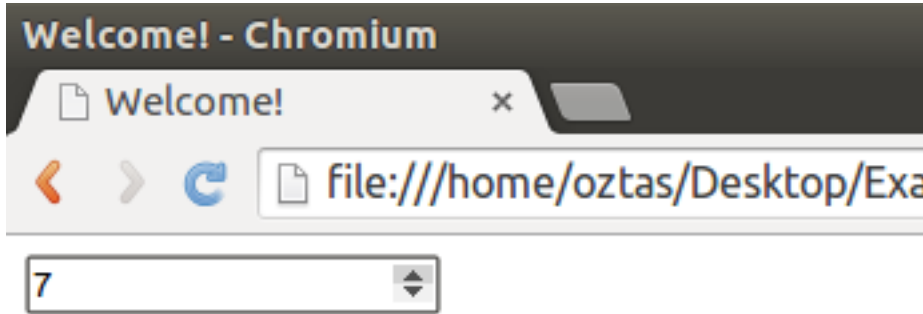
6.1.6 number

number input alanı ile, kullanıcının daha kolay bir şekilde number yani sayı girilmesi sağlanmış olur.

number input alanının kullanımı aşağıdaki gibidir.

```
<input type="number" name="number" />
```

number input alanının görünüşü aşağıdaki gibi olacaktır.



number input alanının farklı attribute (özellikleri) vardır. Örneğin bu number input alanına min ile minimum değeri, max ile maksimum değeri ve step ile değer ilerleme aralığı belirlenir. Aşağıdaki örnek kullanıma bakalım.

```
<input type="number" min="0" max="100" step="5" name="number" />
```

Yukarıdaki örnek kullanımda; kullanıcının girebileceği değer aralığı: 0 -100 ve değer ilerle aralığı: 5

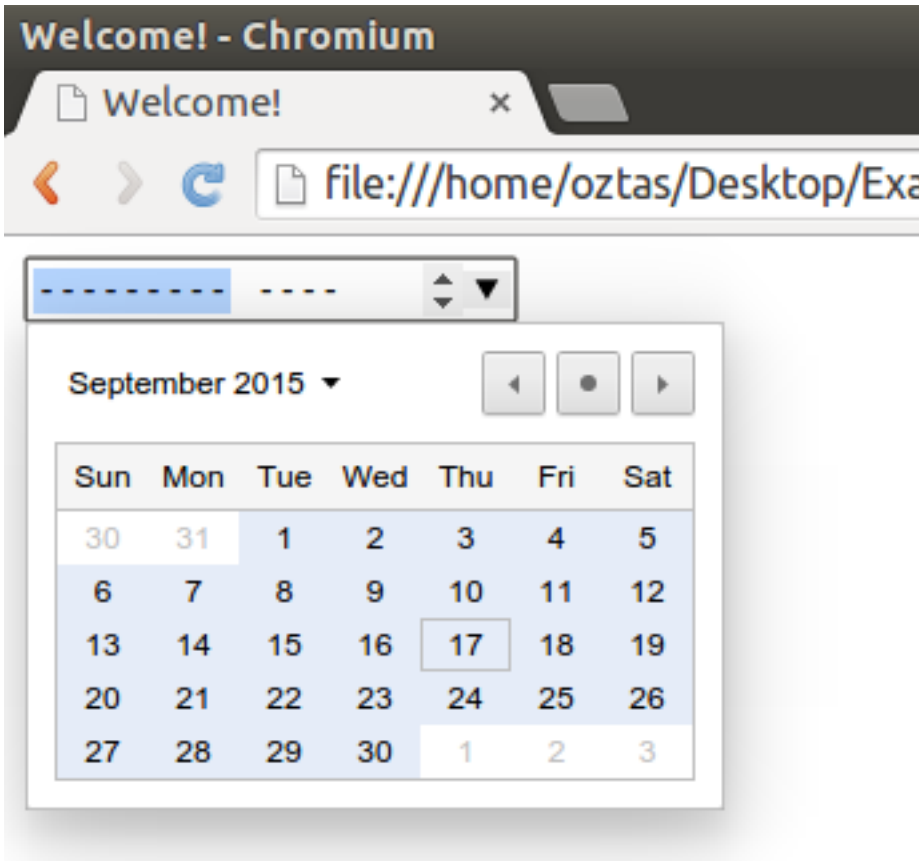
6.1.7 month

month input alanı, month yani ay değerini almak için kullanılan bir input alanıdır. Bu input alanı ile kullanıcıdan doğrudan ay değeri alınabilir.

month input alanının kullanımı aşağıdaki gibidir.

```
<input type="month" name="month">
```

month input alanının görünüşü aşağıdaki gibi olacaktır.



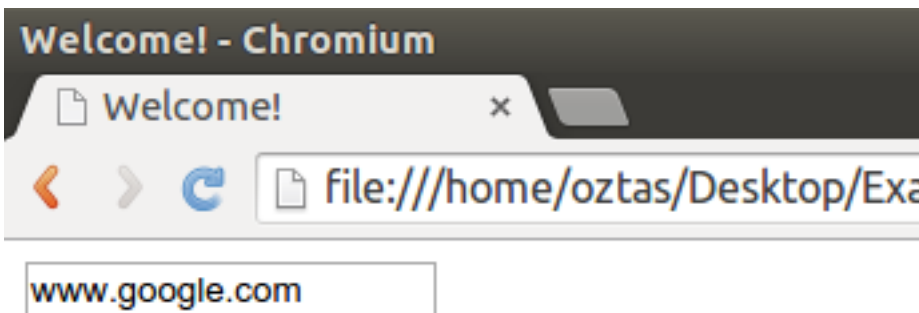
6.1.8 URL

url input alanı ile kullanıcıdan herhangi bir url adı alınmaktadır. Şayet kullanıcı kullandığı tarayıcıdan, daha önce herhangi bir url girmiş ise (illa ki girmiştir.) bu url adresleri de url input alanında kullanıcıya yardımcı olmak için çıkar (url input alanına çift tıklayarak veya aynı isme sahip url yazarken otomatik tamamlama yapılır).

url input alanının kullanımı aşağıdaki gibidir.

```
<input type="url" name="url">
```

url input alanının görünüşü aşağıdaki gibi olacaktır.



6.1.9 range

range input alanı ile bir aralık oluşturur ve bu aralık arasında fare ile range alanın sahip olduğu çubuk kullanılarak değer seçimi yapılabilir. Bunu örneğin şimdiler çöpe attığımız (!) radyolara benzetebiliriz. Örneğin radyonun bir frekans çizgisi olur ve bunu hangi frekansı dinlemek istiyorsak el ile ayarlayabiliriz. İşte range alanı da böyledir.

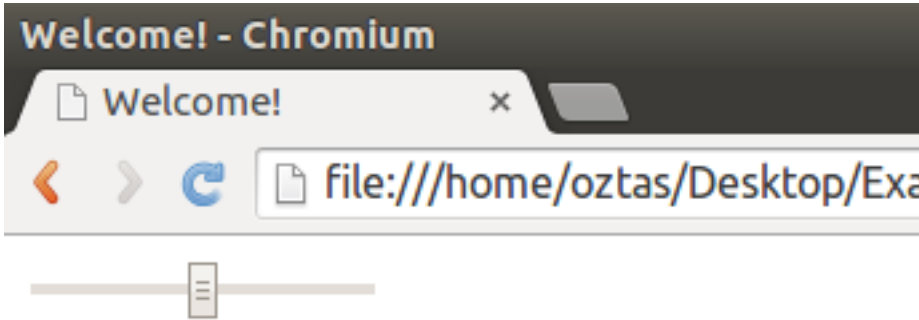
range input alanının kullanımı aşağıdaki gibidir.

```
<input type="range" name="range" />
```

range alanının çeşitli özellikleri vardır. Örneğin; min: minimum değeri, max: maksimum değeri, step: ilerleme aralığı ve value ile şuanki aktif değer belirlenir. Yani range input alanının kullanımı aşağıdaki gibi olacaktır.

```
<input type="range" min="min_value" max="max_value" step="step_value" value="value_for_now" name="range" />
```

range input alanının görünüşü aşağıdaki gibi olacaktır.



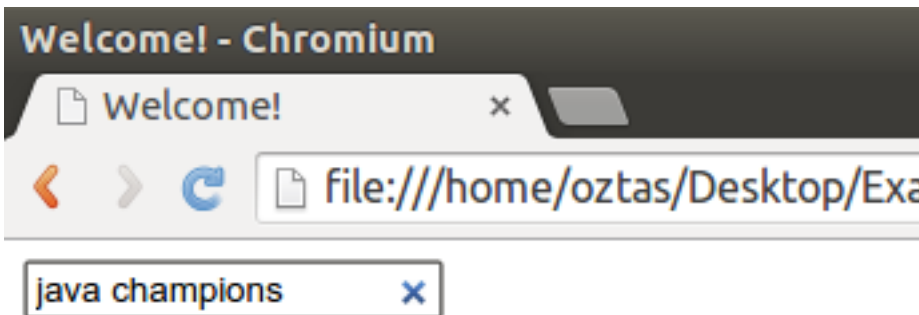
6.1.10 search

search alanı, web site aramasında kullanılan bir alandır. Daha önce kullanıcı tarafından, bu search alanında aranan ifadeler kolaylık olması açısından liste olarak gösterilir.

search input alanının kullanımı aşağıdaki gibidir.

```
<input type="search" name="search">
```

search input alanının görünüşü aşağıdaki gibi olacaktır.



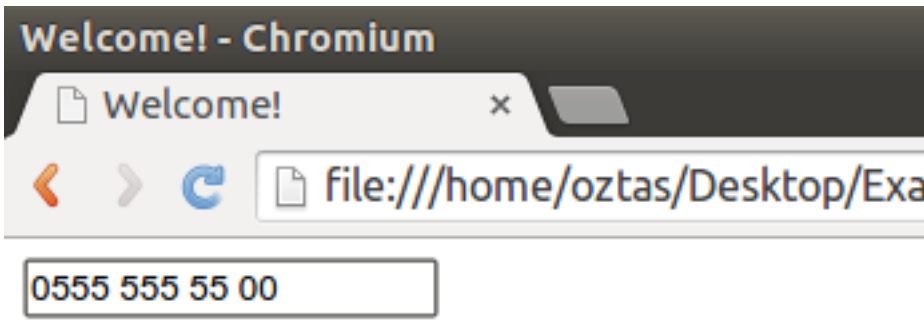
6.1.11 tel

tel alanı, telefon numaraları için kullanılan bir alandır. Kullanıcıdan telefon numarasını alırken kullanılabilir.

tel input alanının kullanımı aşağıdaki gibidir.

```
<input type="tel" name="telephone">
```

tel input alanının görünüşü aşağıdaki gibi olacaktır.



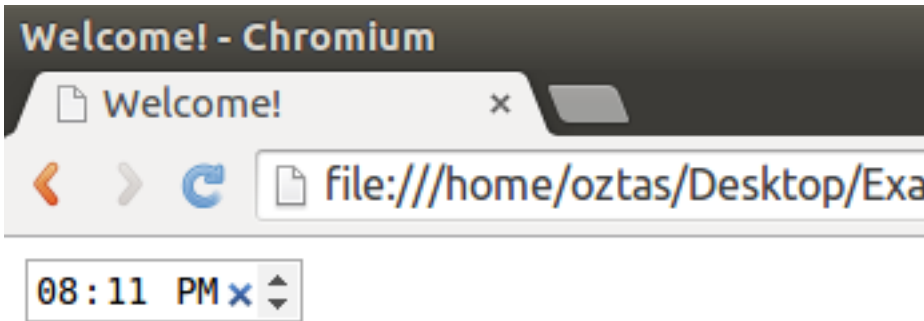
6.1.12 time

time alanı, zamanı almak için kullanılan bir alandır. Alınan zaman değeri, saat için: 1 – 12 ve dakika için: 0 – 59 arasında olmaktadır. Burada da devreye tabiki AM ve PM kavramları girer.

time input alanının kullanımı aşağıdaki gibidir.

```
<input type="time" name="time" />
```

time input alanının görünüşü aşağıdaki gibi olacaktır.



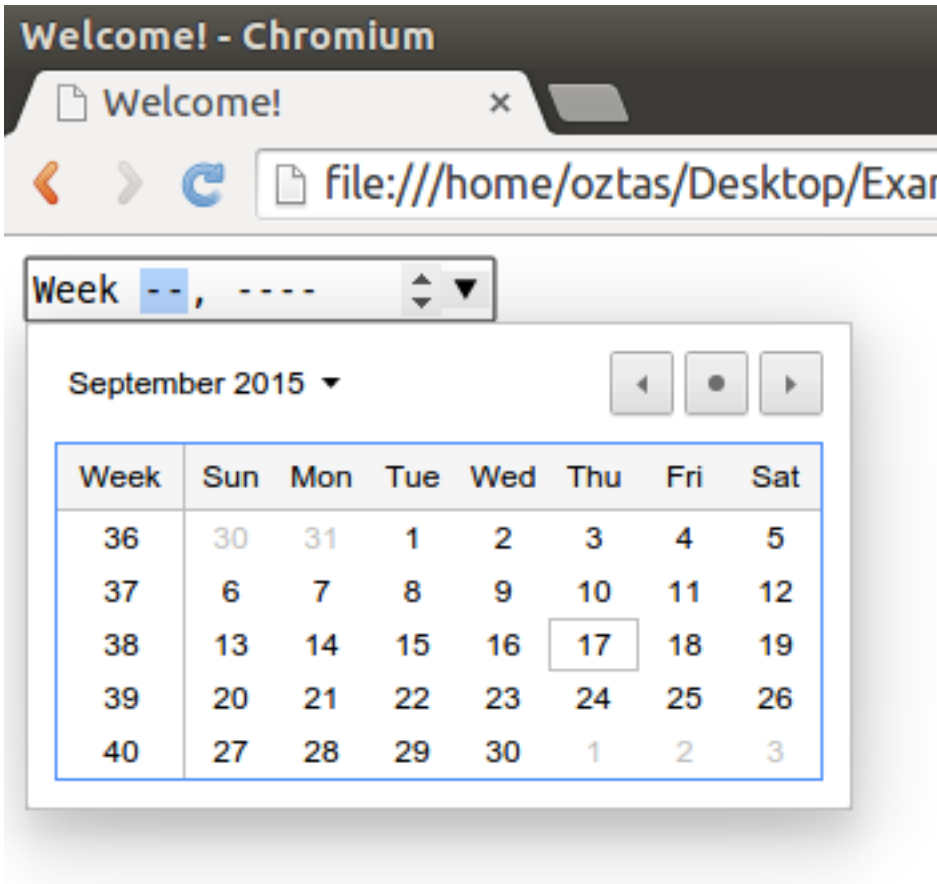
6.1.13 week

week alanı, hafta değerini almak için kullanılır. Alınan hafta değeri 1 – 53 arasında olacaktır.

week input alanının kullanımı aşağıdaki gibidir.

```
<input type="week" name="week" />
```

week input alanının görünüşü aşağıdaki gibi olacaktır.



6.2 New Attributes

6.2.1 autofocus

`autofocus` özelliği, form alanlarının doldurulmasına hangi alandan başlayacağını belirler. Yani sayfa yüklendiği anda `autofocus` özelliğine sahip olan alan otomatik olarak focus yani odaklanılacaktır. Şunu da belirtmek lazım; `autofocus` özelliği yalnızca bir form alanında kullanılmalıdır.

`autofocus` özelliğinin örnek bir form alanı ile kullanımı aşağıdaki gibidir.

```
<input type="text" name="text" autofocus />
```

`autofocus` özelliği herhangi bir form alanında kullanılabilir.

6.2.2 autocomplete

`autocomplete` özelliği, bir çok form alanında kullanılabilir. Bu özellik kısaca; kullanılan form alanında daha önce girilen verileri otomatik olarak doldurmaya veya bir nevi kullanıcının herhangi bir form alanını daha önce girdiği verilerle doldurmasına yardımcı olur.

`autocomplete` özelliği iki değer alabilir. Bunlar; `on` ve `off`.

`autocomplete` özelliğinin örnek kullanımı aşağıdaki gibidir.

```
<input type="search" name="search1" autocomplete="on">
<input type="email" name="email" autocomplete="off">
```

`autocomplete` özelliğini kullanabilen form alanları:

color - password - url - tel - search - text

6.2.3 form

Bildiğimiz gibi form bir alandır ve bu alan içerisindeki alanlar veya veriler; GET ve POST ile taşınır. Yani basit bir form yapısı aşağıdaki gibi olabilir.

```
<form action="#" method="post">
<input type="text" name="kullaniciAdi">
<input type="password" name="sifre">
<input type="submit" value="Login">
</form>
```

Yukarıdaki örnek form alanında da görüldüğü gibi bir verinin taşınabilmesi için form alanının içerisinde olmalı(ydı)dır. HTML 5 ile gelen form özelliği ile herhangi bir form alanı form elementlerinin içerisinde olmasa bile o form elementinin içerisine dahil olabilmektedir.

form özelliğinin örnek kullanımı aşağıdaki gibidir.

```
<form action="#" method="post" id="loginForm">
<input type="text" name="kullaniciAdi" />
<input type="password" name="sifre" />
</form>
<!-- form alanı dışında olan alan -->
<input type="text" name="gizliSoru" form="loginForm">
```

Yukarıdaki örnek kullanımda da görüldüğü gibi form alanının id değeri ve form alanının dışındaki alanın form özelliğiyle belirtilmiş değeri aynı olmalıdır. Bu sayede tarayıcı, bu alanın form alanına ait olduğunu anlayacaktır.

form özelliği herhangi bir form alanında kullanılabilir.

6.2.4 min & max

Bazı form alanlarında min ve max değerlerini kullandık. Örneğin; number ve range form alanlarında kullandık. Dah önce de belirttiğimiz gibi min: minimum ve max: maksimum değerleri belirtmekte ve kısaca bir aralık oluşturmak için kullanılır.

Hem min hemde max değerlerini birlikte kullanması zorunlu değildir. Herhangi istediğiniz alt veya üst değeri belirlenebilir.

min & max özelliğini kullanabilen form alanları:

date - datetime - datetime - local - range - number - time - week - month

6.2.5 multiple

multiple özelliği, belirtilen alanın birden fazla değer alabileceğini belirtir ve bu değerler comma (virgül ',') işareti ile birbirlerinde ayrılır. multiple özelliği genellikle file form alanı ile birlikte kullanılır.

multiple özelliğinin örnek kullanımı aşağıdaki gibidir.

```
<input type="file" name="file" multiple>
```

multiple özelliğini kullanabilen form alanları:

mail - file

6.2.6 pattern

pattern özelliği ile bir anlamda Regular Expression özelliğini yakalamış oluruz. Yani pattern özelliği ile belirtilen aralıkta değer girilmesi sağlanır. Aşağıdaki örnek kullanımı verilmiş olan pattern özelliğine bakalım.

```
<input type="text" name="kullaniciAdi" pattern="[A-Z][0-9]">
```

pattern özelliğini kullanabilen form alanları:

email - search - tel - url - text - password

6.2.7 placeholder

placeholder özelliği ile form alanlarına bir açıklama veya başka herhangi bir metinsel ifade yazabiliriz. placeholder özelliğini value özelliği ile karıştırılmamalıdır. Çünkü value özelliği doğrudan alana bir değer atarken placeholder ise bu alana sadece bir açıklama veya tanım yapmakla yükümlüdür.

placeholder örnek kullanımı aşağıdaki gibidir.

```
<input type="text" name="kullaniciAdi" placeholder="Kullanıcı adı">
```

placeholder özelliğini kullanabilen form alanları:

email - search - tel - url - text - password

6.2.8 required

required özelliği ile herhangi bir form elementi kapsama alanı içerisindeki form alanına, kullanıcıların herhangi bir değer girmediği durumda bu form elementi formu submit (yollamak) etmez.

required özelliğinin örnek kullanımı aşağıdaki gibidir.

```
<input type="text" name="kullaniciAdi" required>
```

required özelliğini kullanabilen form alanları:

email - search - tel - url - text - password - date - number -
checkbox - radio - file

6.2.9 step

step özelliğini daha önceki form alanlarında görmüştük. step özelliği ile herhangi sayısal bir değer tutabilen form alanlarının ilerleme aralığı belirlenebiliyor. Örneğin ilerleme aralığı 5 olarak belirlediğimiz zaman form alanının değerleri -5 veya +5 oranında artıp / azalacaktır.

step özelliğinin örnek kullanımı aşağıdaki gibidir.

```
<input type="number" name="herhangiBirSayi" step="5">
```

step özelliğini kullanabilen form alanları:

```
datetime - datetime-local - time - week - month - number - range
```


BÖLÜM 7:

Drag-N-Drop

HTML 5 ile gelen en önemli özelliklerden birisi de Drag-N-Drop (Sürükle - Bırak) özelliğidir. Bu özellik ile web sayfasında yer alan nesneler bir yerden alınıp daha önce belirlenen herhangi bir alana konumlandırılabilir.

Daha önce hiç puzzle oynadınız mı bilmiyorum. Örneğin bir bütünün parçası olan resimleri doğru konumlara yerleştirerek; ana parçaya ulaşmaya çalışırsınız. Hatta marketlerde veya özellikle kitapçılarda satılan 500, 1.000 hatta 10.000 parçalı puzzle'lardan bahsediyorum. İşte böyle bir oyunu daha doğru böyle bir mantığı HTML 5 ile kolayca yapabilirsiniz.

Drag-N-Drop özelliğini gerçekleştirebilmek için JavaScript'ten yardım almamız gerekir. JavaScript'in event listeners özelliğini kullanarak, Drag-N-Drop işlemini gerçekleştireceğiz.

Herhangi bir resmi, kendi konumundan belirlenen herhangi bir alana taşıma işlemini gerçekleştirelim. Bu işlemimizi bir örnek üzerinde gösterelim. Açıklamalarımızı örneğimiz üzerinde yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
</head>
<style>
/*
    Belirtilen nesnenin ilk konumunu ve son konumunu,
    CSS ile belirleyelim.
    Bu iki alanın büyüklüklerinin aynı olması dikkat edelim.
    Bu alanlar bir container (taşıyıcı) görevi görecek.
    Bu alanlar: birinci ve ikinci div
*/
#divOne {width:300px;height:200px;padding:10px;border:1px solid red;}
#divTwo {width:300px;height:200px;padding:10px;border:1px solid red;}
</style>
<script type="text/javascript">

    // drop-n-drag özelliğini aktif edelim.
    function allowDrop(event){
        event.preventDefault();
    }

    // drag işlemi başladığı zaman bu fonksiyon aktif olacaktır.
    function drag(event){
        // aşağıdaki kod satırı ile sürüklenene nesnenin özelliklerini alalım.
        event.dataTransfer.setData("Text",event.target.id);
    }

    // drag işlemi başlayıp bittiği zaman (drop) bu fonksiyon aktif olacaktır.
    function drop(event){
        // aşağıdaki kod satırı ile sürüklenene nesnenin özelliklerini alalım
```

```

        // ve belirtilen konuma yerlestirelim.
        event.preventDefault();
        var data=event.dataTransfer.getData("Text");
        event.target.appendChild(document.getElementById(data));
    }

</script>
</head>
<body>

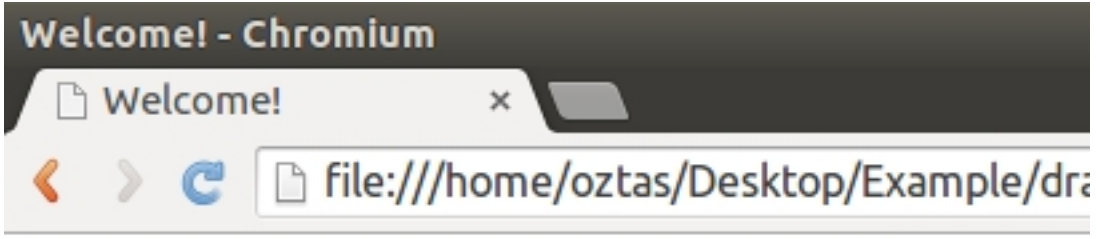
<!-- bir div alanı tanımlayalım. -->
<div id="divOne" ondrop="drop(event)" ondragover="allowDrop(event)">
<!-- bu div alanına bir resim dosyası ekleyelim -->
<!--
    resmin draggable özelliği true olmalı.
    Yani bu resim drag (sürük) edilebilir, demek istedik.
    ondragstart özelliğinde drag(event) özelliğini aktif ettik.
    Sürüklemeye işlemi başladığı zaman bu JavaScript fonksiyon aktif olacaktır.
-->

</div>
<!-- ikinci bir div alanı tanımlayalım -->
<!--
    bu div alanı resim drop (bırak) edildiği zaman
    resmi taşıyacak olan alan olacaktır.
    Yani kısaca: ilk div resmimizi tutacak,
    ikinci div alanı ise resim sürüklenip bırakıldığı zaman
    resmimizi tutacak.
    Sürüklenip bırakıldığı ondrop özelliğindeki drop(event)
    JavaScript fonksiyon aktif olacaktır.
-->
<div id="divTwo" ondrop="drop(event)" ondragover="allowDrop(event)"></div>

</body>
</html>

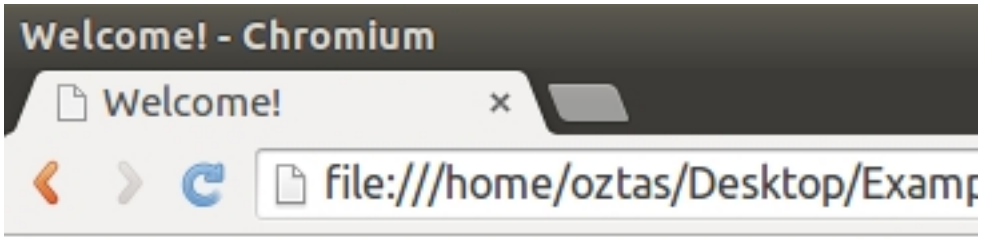
```

Yukarıdaki bolca açıklamalarıyla verdiğimiz örneğimizin ilk halinin ekran çıktısını alalım.



Yukarıdaki ekran alıntısında da görüldüğü gibi ekran çıktımız bu şekilde olacaktır. Şimdi resmimi ikinci alana sürükleyip bırakalım.

Drag – N – Drop işleminden sonra ekran çıktımız aşağıdaki gibi olacaktır.



Örneğimizde verdiğimiz yapımızı kullanarak çok çeşitli yapılar oluşturabilirsiniz.

BÖLÜM 8:

Geolocation

Geolocation özelliği ile kullanıcının aktif konumunu öğrenebilmekteyiz. Bu konum değeri coğrafi bir konumdur. Örneğin kullanıcı A şehrinde ise biz bunu geolocation özelliği ile kolayca öğrenebilmekteyiz. Burada dikkat edilmesi gereken nokta; kullanıcının haberi olmadan konumunu almak hem etik değil hemde suçtur. Lakin şuan kullandığımız onlarca uygulama haberimiz olmadan bu işi arkaplanda yapmaktadır.

Geolocation özelliği daha çok Smart Phones (Akıllı Telefonlar) için geliştirilmiş bir uygulamadır. Yani demem o ki akıllı telefonlarda bu özellik oldukça yararlı ve kullanışlıdır. Örneğin herhangi bir web sitesine bağlanan kullanıcının coğrafi konumunda elde edebiliriz burada herhangi bir sıkıntı yok. Tizen mobil işletim sistemi, PhoneGap, Apache Cordova ile uygulamalarda; HTML 5 + CSS + JavaScript ile uygulama yazılabildiği için bu özellik oldukça önemli bir yere sahip olacaktır.

Bu özelliği kullanmak için JavaScript'ten yardım almamız gerekir. Zaten HTML 5'in çıkışıyla birlikte HTML ve JavaScript ayrılmaz bir bütün haline gelmişlerdir.

Bu konum değerleri; IP adresi, GPS, WIFI, RFID ve GSM/CDMA gibi kaynaklar kullanılarak elde edilir. Tabi ki kesin bir konum bilgisini almak her zaman zordur.

Bir kullanıcının 2 farklı şekilde coğrafi konumunu alabiliriz. Bunlardan ilki kullanıcının konumunu bir sefere mahsus almaktır. İkincisi ise kullanıcının anlık konumunu almaktır. Yani kullanıcı konumu değiştirdiği anda yeni konum değerini almaktır.

Geolocation özelliğini kullanmak için: JavaScript ortamında navigator.geolocation nesnesinden yararlanmamız gerekir. navigator.geolocation nesnesinin, yukarıda bahsettiğimiz iki durum içinde ayrı fonksiyonları vardır. Bunlar:

- `getCurrentPosition()`; // bir sefere mahsus alınan coğrafi konum al.
- `watchPosition()`; // kullanıcı konumu değiştiğinde yeni konum al.

Yani;

```
navigator.geolocation.getCurrentPosition();  
navigator.geolocation.watchPosition();
```

Yukarıdaki her iki fonksiyon da çeşitli parametreler alırlar. Bu parametre değerleri için fonksiyonlar tanımlayıp, çıkan sonuca göre bu fonksiyonları işletmemiz gerekir. Bu parametreler aşağıdaki gibidir.

- İşlem başarılı ise sonucu ver. Biz buna kısaca; `getUserPosition` diyelim.
- İşlem hatalıysa, hatayı göster. Buna da kısaca; `showGeoError` diyelim.
- İsteğe bağlı bir parametre (Bunu kullanmak zorunda değiliz).

Kullanıcının `position` (pozisyonu) değerini elde ettik diyelim. Bu değerleri almak için; bu

değerlere karşılık gelen bir takım özellikleri kullanmamız gerekir. Bu özellikler aşağıdaki gibidir.

```
latitude; // coğrafi konumun koordinat
longitude; // coğrafi konumun koordinat
accuracy; // coğrafi konumun metre cinsinde doğruluğu

altitude; // coğrafi konumun yüksekliği
altitudeAccuracy; // coğrafi konumun yükseklik değerinin doğruluğu
heading; // kullanıcının yönü
speed; // kullanıcının saniyedeki, metre cinsinden hızı
```

Not: altitude, altitudeAccuracy ve speed akıllı telefonlarda veya benzer cihazlarda kullanılır, eğer bu tip cihazlarda GPRS varsa.

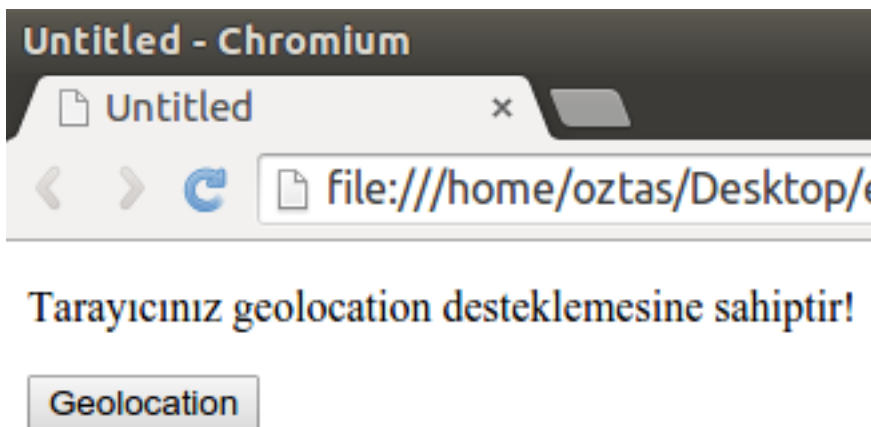
Kullanıcının pozisyonunu elde ederken çeşitli hatalarla karşılaşabiliriz. Bu hatalar aşağıdaki gibidir.

```
PERMISSION_DENIED // izin alınamadı.
POSITION_UNAVAILABLE // konum belirlenemedi.
TIMEOUT // zaman aşımına uğradı.
```

Bu kadar açıklamadan sonra uygulamaya geçelim. İlk olarak tarayıcımız geolocation'ı destekliyor mu buna bakalım. Aşağıdaki JavaScript kodlarımızı yazıp çıktısına bakalım.

```
<script type="text/javascript">
    function getGeolocation(){
        if(navigator.geolocation)
            document.getElementById("query").innerHTML = "Tarayıcınız
geolocation desteklemesine sahiptir!"
        else
            document.getElementById("query").innerHTML = "Tarayıcınız
geolocation desteklemesine sahip değildir!"
    }
</script>
```

Yukarıdaki kodlarımızın ekran çıktısını alalım.



Yukarıdaki ekran alıntısında da görüldüğü gibi tarayıcımız geolocation desteklemesine sahiptir.

Buraya kadar çok şey söyledik. Bu söylediklerimizi bir örnek üzerinde görelim. Örneğimiz aşağıdaki gibi olacaktır.

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
<script type="text/javascript">

    function getGeolocation() {
        var userPosition = document.getElementById("userPosition");
        /*
            Kullanıcının konum bilgilerini elde etmeye çalışalım.
            Eğer bu işlem başarılı ise 'success', başarısız değilse
            'error' fonksiyonu çağırılsın.
        */
        navigator.geolocation.getCurrentPosition(success, error);

        // success fonksiyonu: konum bilgilerini alalım
        function success(position) {
            // koordinatları alalım
            var latitude = position.coords.latitude;
            var longitude = position.coords.longitude;
            //aldığımız koordinatları yazdıralım
            userPosition.innerHTML = "Latitude: " + latitude + "Longitude: " +
            longitude + "<br>";

            // Kullanıcının konumunu haritada göstermek için bir img nesnesi
            tanımlayalım.
            var img = new Image();
            // Bu img nesnesi google - map ile dolacak.
            // Kullanıcının konumunu harita üzerinde işaretleyelim.
            // Eğer zoom değerini arttırırsanız daha sağlıklı bir harita ile
            karşılaşırsınız.

            img.src = "https://maps.googleapis.com/maps/api/staticmap?center=" +
            latitude + "," + longitude + "&zoom=10&size=600x600&sensor=true";

            // Kullanıcının harita üzerindeki değerini gösterelim
            userPosition.appendChild(img);
        };

        // error fonksiyonu: konum bilgileri çeşitli sebeplerle alınamıyor.
        function error() {
            userPosition.innerHTML = "Konum bilgisine ulaşılamadı!";
        };

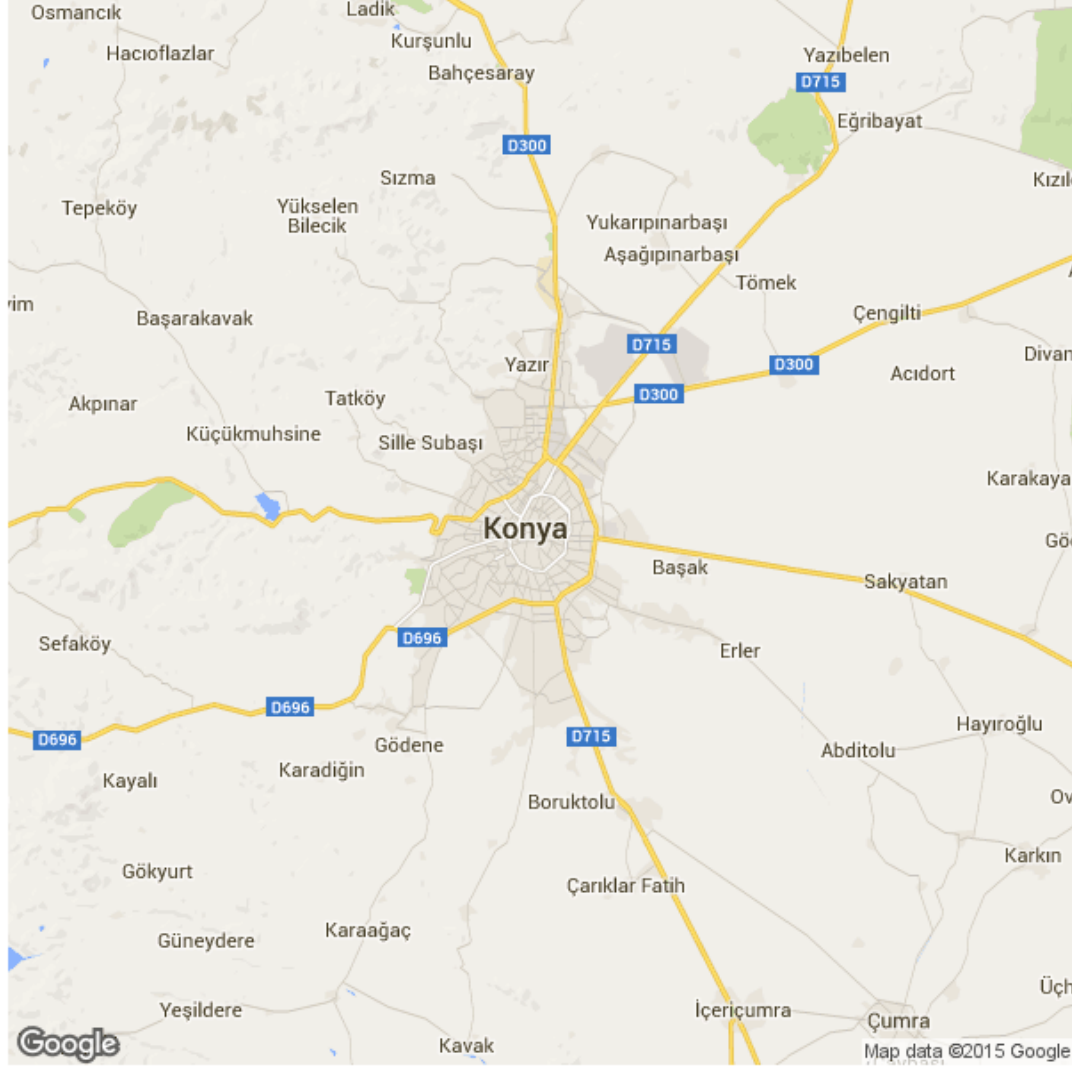
        /*
            Kullanıcı konumuna ulaşmak internet bağlantınıza göre biraz zaman
            alabilir.
            0 yuzden kullaniciya beklenmesini soyleyelim
        */
        userPosition.innerHTML = "<p>Bekleniyor...</p>";
    }
</script>
</head>
<body>
<output id="userPosition"></output>
<input type="button" name="button" value="Geolocation" onclick="getGeolocation()"/>
</body>
</html>

```

Yukarıdaki örneğimizde açıklamalarımızı kodlar arasına yazdık. Şimdi konum bilginizi alalım bakalım doğru bir sonuca ulaşabiliyor muyuz. Burada hatırlamakta yarar var. tarayınız sizden onay isteyecektir.

Bu onayı vermediğiniz sürece konum bilginiz gösterilmez.

Latitude: 37.872175999999996Longitude: 32.505259099999996



Geolocation

Yukarıdaki ekran alıntısında da görüldüğü gibi konum değerimizi elde ettik.

Bu örneğimizde kullanıcının konumunu sadece bir sefere mahsus almadık. Ama isterseniz watchPosition() fonksiyonu ile kullanıcıyı takip edebilirsiniz.

BÖLÜM 9:

Local Storage

Bu bölümde HTML 5 ile gelen Local Storage kavramı üzerinde duracağız. Peki Local Storage nedir? Local Storage belli bilgilerin tarayıcıda saklanmasını sağlayan bir yapıdır. Örneğin kullanıcının session (oturum) bilgileri v.s gibi. Bilindiği gibi bu bilgiler daha önce cookies (çerez)'de saklanırdı. Bu da pek sağlıklı bir yöntem değildi. Çünkü kullanıcının bilgisayarına sızma ve kullanıcı bilgilerini çalma çok yaygındı. Lakin Local Storage kavramı ile bilgiler kullanıcının tarayıcısında saklanır ve cookies'e göre çok daha güvenlidir.

2 tip Local Storage versiyonu vardır. Bunlar:

- Local Storage
- Session Storage

Local Storage yani JavaScript fonksiyonu olan `window.localStorage`, kaydedilen bilgileri, belirtilmeyen herhangi bir tarihe kadar saklayabilir. Tabi istenilen zamanda bu bilgiler silinebilir.

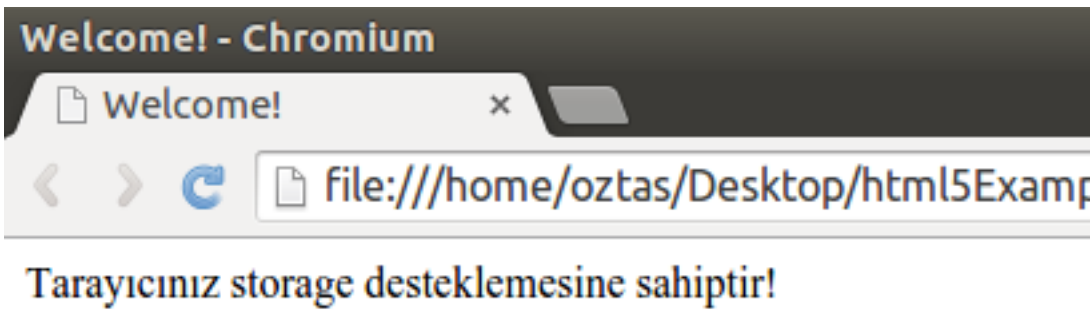
Session Storage yani bir diğer JavaScript fonksiyonu olan `window.sessionStorage`, saklanmak istenilen bilgileri yalnızca bir oturum için saklar. Yani kullanıcı web sayfasını kapattığı zaman ya da tarayıcıyı kapattığı anda bu bilgiler hemen tarayıcı belleğinden silinirler.

Local Storage kavramı JavaScript ile gerçekleştirilebilir. Bu konuda JavaScript ile çalışacağız. Bir diğer konudan bahsetmek isterim. Bu tarayıcıda depolanan bilgiler aynı yerde saklanırlar. Hangi web adresine bağlandığınızın bir önemi yoktur. Hepsini tek bir adreste saklanırlar ve her birinde farklı web adreslerine ait olduklarını bilirler. Yani A sitesi için saklanan bilgiler B sitesi için farklıdır ve bu web adresinde bilinmezler.

Local Storage ve Session Storage kavramlarına geçmeden önce tarayıcımızın bu iki kavramı destekleyip desteklemediklerine bir bakalım. Aşağıdaki JavaScript kodlarımızı yazalım.

```
<script type="text/javascript">
var udef = "undefined";
if(typeof Storage != udef)
document.write("Tarayıcınız storage desteklemesine sahiptir!");
else document.write("Tarayıcınız storage desteklemesine sahip değildir!");
</script>
```

Yukarıdaki JavaScript kodumuzu web sayfamıza ekleyelim ve tarayıcı da çıktısına bakalım. Bende çıkan sonuç aşağıdaki gibidir.



Yukarıdaki ekran çıktısında görüldüğü gibi tarayıcım olan Chromium Storage'i desteklemektedir.

Şimdi sırasıyla Local Storage ve Session Storage kavramlarına ayrı ayrı olarak bakalım.

9.1 Local Storage

ilk Local Storage ile başlayalım. Bölümümüzün başında da belirttiğimiz gibi Local Storage ile tarayıcımızda veri saklayabilmekteyiz. Bu veriyi, silmediğimiz sürece saklı kalacaktır ve istediğimiz zaman tekrar bu veriye erişebileceğiz.

Local Storage özelliğini kullanarak tarayıcımıza veri saklama işlemini gerçekleştirelim. Bu işlemi iki farklı yolla gerçekleştirebiliriz.

Birinci yöntem aşağıdaki gibidir.

```
localStorage.setItem(key, value);
```

Yukarıdaki yöntemde key kelimesi yerine; saklayacağımız bilginin adını vermeliyiz. Bu bilgi eğer bir kelime ise tırnak (" " veya ' ') işaretleri arasına, şayet sayı ise bu sayıyı doğrudan buraya yazmalıyız. value kelimesi yerine de saklayacağımız bilgimizi yazmalıyız. Yani value alanına yazacağımız değer, saklanacak olan bilginin bir başlığı veya adı olarak tabir edebiliriz. key alanına yazacağımız metinsel / sayısal kural burada da geçerlidir. setItem() fonksiyonu ile de bilgiyi saklamaktayız.

İkinci yöntem aşağıdaki gibidir.

```
localStorage.key = value;
```

Yukarıdaki örnek kullanımımıza bakalım. Bu kullanımımızda key yerine bilginin başlığını veya adını yazmalıyız ve value yerine de bu bilginin içeriğini yazmalıyız. Yani burada setItem() fonksiyonu yerine doğrudan bilginin başlığını yazarak bu bilginin içerik açıklamasını yapıyoruz.

Aşağıdaki örnek kullanımlara bakalım.

```
localStorage.setItem("bilgi1", "Türkiye'nin başkenti Ankara'dır.");  
localStorage.bilgi2 = "Türkiye'den selamlar!";
```

Yukarıdaki örnek kullanımda; bilgi1 ve bilgi2 adındaki iki tane key değerimiz var ve bunlarda saklanan değerler var.

Tarayıcımızda verileri sakladık peki bu değerlere sonra nasıl erişebileceğiz? Şimdi bu sorumuza yanıt arayalım. İki tane kullanım şekli verdik. İlk olarak ikinci kullanım şeklinden başlayalım ve bu değeri geri nasıl çağırabiliriz buna bakalım. İkinci kullanımdan başlamamız tamamen kolaylık açısından. Aşağıdaki örneğimizi inceleyelim.

```

<script type="text/javascript">

// verimizi, bilgi2 adıyla tarayıcımızda saklayalım.
localStorage.bilgi2 = "Türkiye'den selamlar!";
//verimizi çağıralım.
var veri2 = localStorage.bilgi2;
/*
veya document.write(localStorage.bilgi2);
olarak doğrudan ekrana yazdırabiliriz.
*/

</script>

```

Yukarıdaki örneğimizde görüldüğü gibi; verimizi yine kendi adını kullanarak çağırabilmekteyiz.

Birinci kullanım şekline dönelim ve sakladığımız verimizi geri nasıl çağırabiliriz, şimdi buna bakalım. Aşağıdaki örneğimizi inceleyelim.

```

<script type="text/javascript">

// verimizi, bilgil adıyla tarayıcımıza saklayalım.
localStorage.setItem("bilgil", "Türkiye'nin başkenti Ankara'dır.");
// verimizi çağıralım.
var veri2 = localStorage.getItem("bilgil");
/*
veya document.write(localStorage.getItem("bilgil"));
olarak doğrudan ekrana yazdırabiliriz.
*/

</script>

```

Yukarıdaki örneğimizi incelediğimizde karşımıza `getItem()` kavramı çıkmaktadır. İşte bu fonksiyon yardımıyla, `setItem()` fonksiyonu ile kaydettiğimiz verimizi alabilmekteyiz. Ayrıca `getItem()` fonksiyonu ikinci kullanım içinde geçerlidir. Hangisini kullanmak isterseniz.

Verilerimizi tarayıcımızda sakladık ve çağırdık. Peki bu veriyi daha sonra silmek istersek? İşte burada da karşımıza `removeItem()` kavramı çıkmaktadır. Aşağıdaki örneğimize bakalım.

```

<script type="text/javascript">

// verilerimizi kaydedelim.
localStorage.setItem("bilgil", "Türkiye'nin başkenti Ankara'dır.");
localStorage.bilgi2 = "Türkiye'den selamlar!";

// verilerimizi alalım.
veri1 = localStorage.getItem("bilgil");
veri2 = localStorage.bilgi2;

// verilerimizi silelim.
localStorage.removeItem("bilgil");
localStorage.removeItem("bilgi2");

</script>

```

Yukarıdaki örneğimizde `removeItem()` fonksiyonunun kullanımını görmekteyiz. `removeItem()` fonksiyonu ile bu kaydettiğimiz verileri silmediğimiz zaman bu veriler tarayıcımızda hep kalacaktır. Buna dikkat edelim.

Buraya kadar olan ifadelerimizi toplamak ve büyük resmi görmek için bir örnek yapalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
<script type="text/javascript">

    // verilerimizi kaydedelim.
    localStorage.setItem("bilgi1", "Türkiye'nin başkenti Ankara'dır.");
    localStorage.bilgi2 = "Türkiye'den selamlar!";

    // verilerimizi alalım.
    veril = localStorage.getItem("bilgi1");
    veri2 = localStorage.bilgi2;

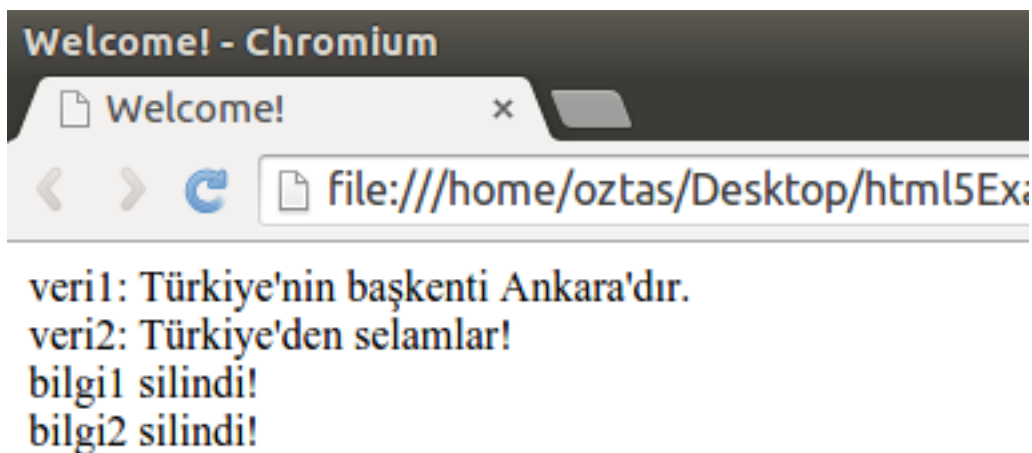
    //verilerimizi yazdıralım.
    document.write("veri1: " + veril + "<BR>");
    document.write("veri2: " + veri2 + "<BR>");

    try{
        // verilerimizi silelim.
        // birinci veri
        localStorage.removeItem("bilgi1");
        document.write("bilgi1 silindi!" + "<BR>");
        //ikinci veri
        localStorage.removeItem("bilgi2");
        document.write("bilgi2 silindi!");
    }catch(err){
        document.write("silinemedi!");
    }

</script>
</head>
<body>

</body>
</html>
```

Yukarıdaki örneğimizin ekran çıktısını alalım.



Yukarıdaki örnek ekran alıntısında da görüldüğü gibi işlemlerimizi başarıyla gerçekleştirdik.



Local Storage alanında saklanan bilgiler metinsel formattadır. Yani `String` olarak saklanır. Eğer herhangi bir sayısal değeri kaydedip sonra geri çağırıp işlem yaptırmak isterseniz `Type Casting` (Tip Dönüşümü) yapmanız gerekir.



`localStorage` alanında kaç veri saklandığını öğrenmek isterseniz: `length` fonksiyonunu kullanabilirsiniz.

```
localStorage.length;
```



`localStorage` alanındaki saklı bulunan tüm verileri silmek isterseniz: `clear()` fonksiyonunu kullanabilirsiniz.

```
localStorage.clear();
```

9.2 Session Storage

`Session Storage` konusunu, bu bölümün başında açıklamıştık. `Session Storage` sadece bir `session` (oturum) için geçerlidir. Eğer kullanıcı sayfayı kapatırsa veya tarayıcıyı kapatırsa bu `Session Storage` alanındaki veriler silinecektir.

Local Storage bölümünde söylediklerimizin hepsi `Session Storage` içinde geçerlidir. Her ikisi de aynı fonksiyonlara sahiptir. Çünkü ikisinin de temel olarak yaptıkları iş aynıdır. Değişen sadece nesne isimleridir. `Session Storage` için `sessionStorage` nesnesini kullanırız.

Local Storage konusunda yaptığımız örneği `Session Storage` için uyarlayalım. Örneğimiz aşağıdaki gibi olacaktır.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
<script type="text/javascript">

    // verilerimizi kaydedelim.
    sessionStorage.setItem("bilgi1", "Türkiye'nin başkenti Ankara'dır.");
    sessionStorage.bilgi2 = "Türkiye'den selamlar!";

    // verilerimizi alalım.
    veri1 = sessionStorage.getItem("bilgi1");
    veri2 = sessionStorage.bilgi2;

    //verilerimizi yazdıralım.
    document.write("veri1: " + veri1 + "<BR>");
    document.write("veri2: " + veri2 + "<BR>");

    try{
        // verilerimizi silelim.
        // birinci veri
        sessionStorage.removeItem("bilgi1");
        document.write("bilgi1 silindi!" + "<BR>");
        //ikinci veri
        sessionStorage.removeItem("bilgi2");
        document.write("bilgi2 silindi!");
    }catch(err){
```

```
        document.write("silinemedi!");  
    }  
</script>  
</head>  
<body>  
  
</body>  
</html>
```

BÖLÜM 10:

Web Workers

Bu bölümde Web Workers kavramı üzerinde duracağız. Web Workers nedir? Ne işe yarar? Nasıl kullanılır? v.s gibi soruların cevaplarını arayacağız. İlk olarak Web Workers nedir sorusundan başlayalım. Örneğin bir web sayfası düşünelim. Bu web sayfasında; JavaScript kodları sayfa içine embedded (gömülü) bir şekilde de yer alabilir, external (harici) bir dosya içinde de yer alabilir ve işlevlerini yerine getirebilirler.

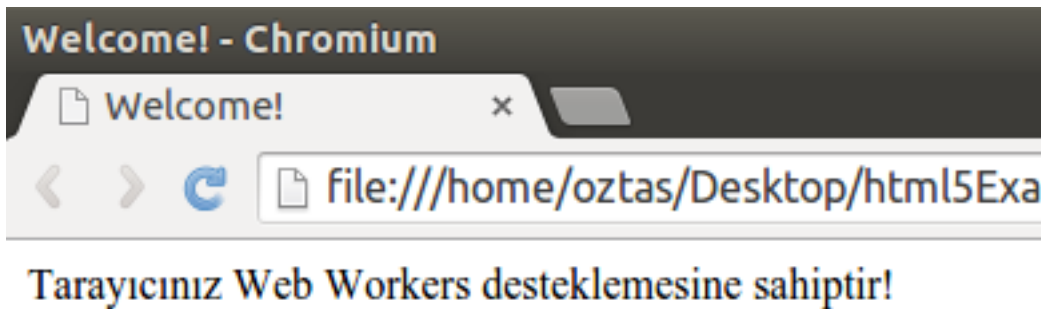
JavaScript kodlarını çalıştırdığımız anda web sayfası bir süre kesintiye uğrar. Bu kesintiye uğramasının sebebi; JavaScript kodlarının işlevini yerine getirmesini beklemesidir. Şuan kullandığımız bilgisayarlarımız çok hızlı. Örneğin benim laptopum 7 yıllık olmasına ve garip sesler çıkarmasına (Traktör sesine benzer bir ses) rağmen oldukça stabil ve hızlı çalışıyor. O sebepten web sayfalarında bu meydana gelen kesintiye anlayamıyoruz bile. Şayet JavaScript kodları sonsuz döngüye v.s girmemişse. İşte burada Web Workers devreye giriyor.

Web Workers'lar web uygulamalarında arka planda sessizce çalışırlar, haberiniz olmaz. Web sayfasını herhangi bir kesintiye uğratmazlar ve herhangi bir performans kaybı yaşatmazlar. Sonsuz döngü v.s girseler bile. Bu sebepten ötürü kullanımları web sayfasında herhangi bir olumsuz etkiye sebebiyet vermez.

Web Workers'lar harici JavaScript dosyalarıdır. İstemediğimiz sürece çalışmazlar veya çalışmayı bırakmazlar. Web Workers kavramını anlamak için çalışmalara başlayalım. İlk olarak tarayıcımız Web Workers'ı destekliyor mu bakalım. Aşağıdaki JavaScript kodlarımızı yazıp çalıştıralım.

```
<script type="text/javascript">
var udef = "undefined";
if(typeof Worker != udef)
document.write("Tarayıcınız Web Workers desteklemesine sahiptir!");
else document.write("Tarayıcınız Web Workers desteklemesine sahip değildir!");
</script>
```

Yukarıdaki kodlarımızın ekran çıktısını alalım.



Ekran alıntısında da görüldüğü gibi tarayıcım Web Workers desteklemesine sahiptir. Tarayıcınız çok eski

değilse muhtemelen bu desteklemeye sahiptir. Bu testini yapmış olduğum tarayıcı; Chromium. Fakat Chrome veya Chromium'da şöyle bir durum var: eğer local'de çalışıyorsanız, Web Workers'ı tam olarak desteklemiyorlar. Aslında destekliyorlar lakin Web Workers dosyalarının yaptıkları işlemleri göstermiyorlar. O yüzden çalıştığınız ortamı bir Server'e Upload (Yükleme) yapmanız gerekiyor. Eğer bir web alanınız varsa oraya yüklemeniz gerekiyor. Ya da en kısa yolla: bilgisayarınızda kurulu olan herhangi bir local'de çalışan server'da deneyebilirsiniz. Örneğin bende hem PHP Server hemde Apache Tomcat var. İkisinde de deneyebilirim. Benim tercihim daha basit olduğu için PHP Server.

Şimdi yapmamız gereken harici bir JavaScript dosyası oluşturmak. Bu JavaScript dosyası; her ne yapmak istiyorsak o işlemi gerçekleştirecek olan yapı. Aşağıdaki örneğimize bakalım.

```
//web worker: userwebworker.js
var userName = "emrecan-oztas";
var userPassword = "123456";
// mesaj gönderelim
postMessage(userName + " " + userPassword);
```

Yukarıdaki kodlarımızı harici bir JavaScript dosyası olarak hazırlayıp kaydedelim. JavaScript dosyamızın adı: userwebworker.js

Yukarıdaki kodlarımızı çok basit tuttuk. İki tane değişken tanımladık ve bu değişkenlere değer ataması yapıp istek geldiği zaman göndereceğiz. Bu gönderme işlemi postMessage() fonksiyonu ile gerçekleşmektedir. Bu fonksiyonu return gibi görebilirsiniz.

Gelelim ana formumuzu yazmaya. Bu formumuz bizim web sayfamız olacak. Bir Web Worker nesnesi oluşturacağız ve hazırladığımız JavaScript kodlarıyla iletişime geçeceğiz. Aşağıdaki örnek web sayfamıza bakalım.

```
<!-- web sayfası: userindex.html -->
<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">
<script type="text/javascript">

/*
    Bir fonksiyon oluşturalım.
    Bu fonksiyon ile hazırladığımız
    web worker ile iletişime geçeceğiz.
    Bu fonksiyonu da onclick olayla kullanacağız.
*/
function runWebWorker(){

/*
    Bir Worker nesnesi oluşturalım.
    Bu nesnemize hangi harici JavaScript
    dosyasıyla (Web Workers) çalışacağımızı bildirelim.
*/
var myWebWorker = new Worker("userwebworker.js");
/*
    Oluşturmuş olduğumuz nesnemizi dinlemeliyiz.
    Bu dinleme işlemi anlık gerçekleşir.
    Yani nesnemizin bir kulagi Web Worker'da olacak.
    Web Worker'dan gelen, postMessage() ile gönderilenleri
    dinleyecek ve yorumlayacak.
    onmessage fonksiyonu özel bir fonksiyon. Kullanımına dikkat
    edelim.
*/
```



```

myWebWorker.onmessage = function(event){

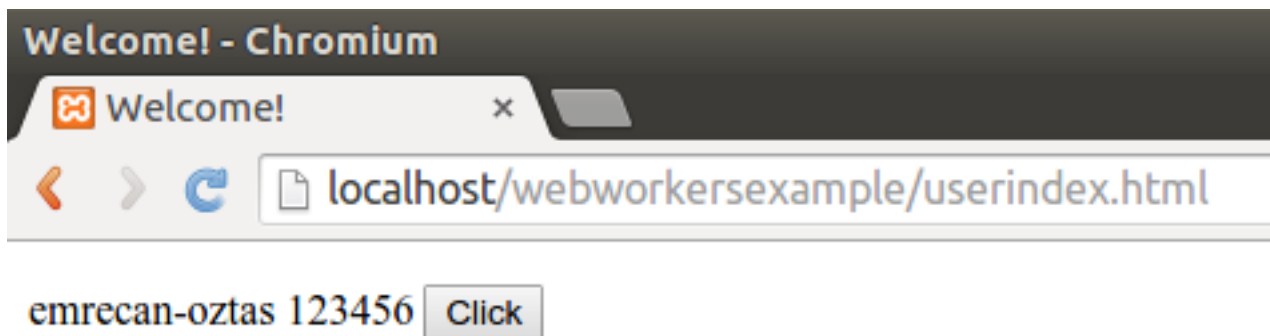
  /*
    Bir degisken tanımlayalım. Web Worker'dan gelen bilgiyi
    almak için.
    Web Worker'dan gelen bilgi 'event.data' olacak.
  */
  var takeValue = event.data;
  // aldığımız bilgiyi DOM ile elemente yerleştirelim.
  document.getElementById("takeValue").innerHTML = takeValue;
}
</script>
</head>
<body>
<p>
<output id="takeValue">Bilgi için tıklayınız!</output>
<!-- onclick olayında runWebWorker() fonksiyonumuzu çağıralım -->
<input type="button" name="button" value="Click" onclick="runWebWorker()"/>
</p>

</body>
</html>

```

Yukarıdaki örneğimizin açıklamasını kodlar arasında yaptık. Bu örneğimizi de `userindex.html` olarak JavaScript dosyamızın yanına konumlandırdık.

Localhost'ta hazırlamış olduğumuz örneğimizi çalıştıralım ve sonuçlarına bakalım.



“Click” butonuna tıkladığımız zaman daha önce hazırladığımız JavaScript kodlarımızdan gelen bilgiyi görebiliyoruz.

Yukarıda örneğimiz çok basit bir örnektir. Bu örneğimizi sadece Web Workers mantığını anlamak için yazdık ve uyguladık. Farklı bir örnek yapalım ve Web Worker özelliğini daha yakından tanımaya çalışalım.

Bu örneğimizde basit bir `kronometre` yapalım. İlk olarak JavaScript dosyamızı oluşturalım. JavaScript dosyamız aşağıdaki gibi olacaktır.

```

// web worker: webworkerkronometre.js

/*
  Sayac degiskeni olarak i adında
  bir degisken tanımlayalım ve 0'a esitleyelim.
*/
var i = 0;

/*

```

```

        Kronometre icin bir fonksiyon tanımlayalım.
        Bu fonksiyon basit olarak i degiskenini
        +1 arttırıp degeri postMessage() ile gonderecek.
    */
function startKronometre() {
    i = i + 1;
    postMessage(i);

    // eger gonderilecek deger 59'a esitse;
    // gonderilecek degeri 0'layalım.
    if(i == 59) {
        i = 0;
    }

    /*
    setTimeout fonksiyonu JavaScript ortamında
    zamanlama fonksiyonudur. yani belli bir isi
    belirtilen zamanda tekrar gerçekleştirir.
    asagidaki satirimizda startKronometre()
    fonksiyonu belirtilen zamanda (100) tekrar tekrar
    cagrilacaktır.
    setTimeout() fonksiyonu yerine setInterval
    fonksiyonunu da kullanabiliriz.
    */
    setTimeout("startKronometre()", 100);
}

// kronometre islemini gerceklestirmek ici
// startKronometre() fonksiyonunu cagiralım.
startKronometre();

```

Şimdi de web sayfamızı oluşturalım.

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Welcome!</title>
<meta charset="utf-8">

<script type="text/javascript">
var myWebWorker;
var dk = 0;
var sn = 0;
var deger = 0;
function runWebWorker(){
    myWebWorker = new Worker("wworker.js");
    myWebWorker.onmessage = function(event){
        deger = event.data;
        if(deger == 59){
            sn = sn +1;
            document.getElementById("sn").innerHTML = sn;
        }

        if(sn == 59){
            dk = dk + 1;
            sn = 0;
            document.getElementById("dk").innerHTML = dk;
        }

        document.getElementById("sl").innerHTML = event.data;
    };
}

```

```
}

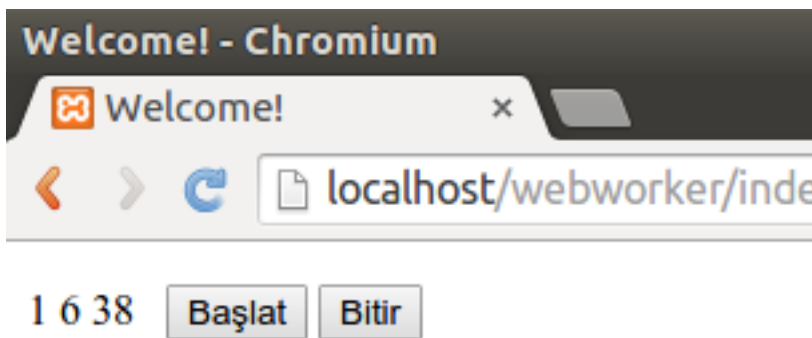
/*
    stopWorker() fonksiyonu, runWebWorker() fonksiyonu ile
    işletmeye başladığımız Web Worker'ı sonlandıracaktır.
    Bu sonlanma işi terminate() fonksiyonu ile gerçekleşir.
    Daha sonra tekrar kullanmak için myWebWorker nesnemizi,
    undefined (değersiz) hale getiriyoruz.
*/

function stopWebWorker(){
myWebWorker.terminate();
myWebWorker = undefined;
}

</script>
</head>
<body>
<p>
<output id="dk">0 </output>
<output id="sn">0 </output>
<output id="sl">0</output>&nbsp;&nbsp;&nbsp;
<input type="button" name="button" value="Başlat" onclick="runWebWorker()" />
<input type="button" name="button" value="Bitir" onclick="stopWebWorker()" />
</p>
</body>
</html>
```

Gerekli açıklamalarımızı, kod aralarında belirttik fakat bir kez daha bahsetmekte fayda var. `myWebWorker` bizim `Worker` nesnemiz. Bu nesne ile oluşturmuş olduğumuz JavaScript kodlarımızı arkaplanda işletiyoruz. İşimiz bittiği zamanda `myWebWorker.terminate()` komutu ile çalışmayı kapatıyoruz. Buna dikkat edelim.

Yukarıdaki örneğimize ait olan ekran çıktısını alalım.



Yukarıdaki örnek ekran alıntısında da görüldüğü gibi oluşturmuş olduğumuz Web Worker yapımız gayet düzgün bir şekilde çalışıyor. Sizde istediğiniz herhangi bir yapıda Web Worker kullanabilirsiniz.

BÖLÜM 11:

New Events

Bu bölümde HTML 5 ile gelen Events yani olaylardan bahsedeceğiz ve olayların genişçe listesini vereceğiz. Öncelikle events veya olay nedir? Bunu açıklamaya çalışalım. Öncelikle bir web sayfamız olduğunu düşünelim. Bu web sayfamıza kullanıcılar gelecektir. İşte kullanıcının bu siteye bağlanmasından, ayrılmasına, linklere, butonlara tıklamasına, input alanlara metin girişleri yapmasına kadar yaptığı her şey bir event yani olaydır.

Kullanıcının yaptığı herşey bir olay. Peki bu olaylar ne işe yarıyor? Olaylar, JavaScript kodlarını tetikler. Yani bir anlamda çağırma mekanizmasıdır. Bu olaylar kümesini JavaScript ortamında yazacağımız fonksiyonlar ile yönetebilmekteyiz. Örneğin kullanıcı bir input alana giriş yapıyorken biz bu olayı keyup ile yakalayabiliriz.

Aşağıda HTML 5 ile olayların listesi ve açıklamaları verilmiştir. Bu olaylar yardımıyla daha fonksiyonel web sayfaları el etmek mümkündür.

Event	Description
offline	Internet bağlantısının kopması durumu
onabort	Sayfa yüklenirken kullanıcının; sayfayı iptal etmesi durumunda oluşan olay
onafterprint	Sayfa yazdırıldıktan sonra ki durum
onbeforeunload	Sayfa yüklenmeden önce oluşan olay
onbeforeprint	Sayfa yazdırılmadan önce ki durum
onblur	Odağın pencereden farklı bir yere kayması
oncanplay	Medya çalınmaya başlanması
oncanplaythrough	Medya çalınmaya başlanıp, sonuna kadar gelinmesi
onchange	Herhangi bir element içeriği (input alanlar) değiştirilmesi durumu
oncontextmenu	contextmenu seçilmesi
ondrag	Herhangi bir elementin sürüklenmesi durumu
ondragend	Herhangi bir elementin sürüklenip bırakılması
ondragenter	Herhangi bir elementin belirtilen hedefe sürüklenmesi
ondragleave	Herhangi bir element belirtilen hedefe sürüklenip bırakıldığında oluşan olay
ondragover	Herhangi bir element sürüklenip belirlenen hedefin üzerine gelindiğinde oluşan olay
ondragstart	Sürükleme işleminin başlaması
ondrop	Sürüklenen elementin bırakılması
ondurationchange	Medya uzunluğunun değiştirilmesi
onemptied	Medya kaynağının alınamaması

onended	Medya çalınmasının bitmesi
onerror	Herhangi bir hata meydana geldiğinde
onfocus	Pencereye odaklanması
onformchange	Form değiştirildiğinde
oninput	Herhangi bir elemente kullanıcı girişi durumu
oninvalid	Herhangi bir element geçersiz olduğu durum
onkeydown	Herhangi bir karakter girildiğinde (input alanlar)
onkeypress	Herhangi bir tuşa basıldığında (input alanlar)
onkeyup	Herhangi bir tuşa basılıp, bırakıldığı durum
onloadeddata	Medya verisi yüklendiğinde
onloadedmetadata	Herhangi bir medya yürütülürken diğer medyanın açılması durumunda oluşan olay
onloadstart	Medya verisi yürütülmeye başlandığında
onmessage	Herhangi bir mesaj başlatılması
onmousedown	Fare ile tıklanması
onmousemove	Fare işaretçisinin hareket ettirilmesi
onmousewheel	Farenin topunun çevrilmesi
onoffline	Web sayfasının kapatılması
ononline	Web sayfasının açılması
onpagehide	Pencerenin gizlenmesi
onpageshow	Pencere görünür olduğunda
onpause	Medya verisinin durdurulması
onplay	Medya verisinin yürütülmeye başlanması
onplaying	Medya verisi yürütülmeye başlandığında
onpopstate	Pencere geçmişi değiştirildiği zaman
onprogress	Tarayıcı, medya verisine ulaştığı zaman
onratechange	Medya verisinin yürütme sayısı değiştiğinde
onreadystatechange	Ready – State durumu değiştirilmesi
onredo	Döküman içeriğindeki değişiklik geri verildiğinde
onresize	Pencere boyutu değiştirildiğinde
onscroll	Kaydırma çubuğu kullanıldığında
onseeked	Medya elementinin arama değeri gerekli uzunlukta değilse ve arama işlemi durdurulmuşsa oluşan olay
onselect	Element seçildiği durum

onstalled	Medya verisi alınırken meydana gelen herhangi bir hata durumunu
onstorage	Herhangi bir döküman yüklendiğinde
onsubmit	Form gönderilmesi
onsuspend	Tarayıcı, medya verisini alırken, alınan verinin durdurulması durumunda oluşan olay
ontimeupdate	Medya yürütülürken, zaman durumunun değiştirilmesi
onundo	Döküman içerisindeki içerik değiştirilip geri alındığı durumda oluşan olay
onunload	Kullanıcının sayfadan ayrılması durumu
onvolumechange	Medya yürütülürken ses değerinin değiştirilmesi
onwaiting	Medya yürütülürken durdurulması ve devam etmek için onay beklemesi durumu

BÖLÜM 12:

HTTP Status Message

Bu bölümde bir web sayfasına istek gönderildikten sonra oluşabilecek hataların kodları ve isimlerinden bahsedeceğiz. Kullanıcılar tarafından, bir web sayfasına istek gönderildiğinde herhangi bir nedenden kaynaklanan sorundan dolayı bir hata mesajı üretilir. Örneğin aramızda HTTP 404 Not Found hatası almayan yoktur herhalde. Bu hata mesajları web dünyasına özeldir. Örneğin farklı programlama dillerinde kendilerine özgü hata mesajları var. Örneğin Java: `NullPointerException`.

Bu bölüm HTML 5 ile gelen yenilikler arasında değil fakat web ortamında çok sık bu hatalarla karşılaştığım için sizinle bu hatalar listesini paylaşmak istedim. Yardımcı olacağına eminim.

Code	Message
100	Continue
101	Switching Protocols
102	Processing
103	Checkpoints
200	OK
201	Created
202	Accepted
203	Non-Authoritative Information
204	No Content
205	Reset Content
206	Partial Content
207	Multi-Status
210	Content Different
300	Multiple Choices
301	Moved Permanently
302	Moved Temporarily
303	See Other
304	Not Modified
305	Use Proxy
306	Switch Proxy
307	Temporary Redirect
308	Resume Incomplete
400	Bad Request
401	Unauthorized
402	Payment Required

403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
407	Proxy Authentication Required
408	Request Timeout
409	Conflict
410	Gone
411	Length Required
412	Precondition Failed
413	Request Entity Too Large
414	Request - URI Too Long
415	Unsupported Media Type
416	Requested Range Not Satisfiable
417	Expectation Failed
422	Unprocessable Entity
423	Locked
424	Method Failure
500	Internal Server Error
501	Not Implement
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version Not Supported
507	Insufficient Storage
511	Network Authentication Required

<http://www.tutorialspoint.com/html5/index.htm>
<http://www.w3schools.com/html/default.asp>
<http://tutorials.jenkov.com/html5/index.html>
<http://www.html-5-tutorial.com/>
<https://www.codeschool.com/courses/front-end-formations>
<http://www.w3resource.com/html5/introduction.php>
<http://www.w3.org/TR/html5/>