VS CODE SHORTCUTS

CONTROL S → KODU KAYDETME

CONTROL + ALT + S → KODUN HEPSİNİ KAYDETME

CONTROL + TAB → BİR SONRAKİ KODA GEÇİŞ YAPAR

CONTROL + W → DOSYA KAPATMA

CONTROL + B → MENU ÇUBUĞUNU AÇIP KAPATIR

CONTROL + P → AÇILAN MENÜDEN DOSYALARI GÖRÜNTÜLEME

CONTROL + C → SEÇİLİ İFADEYİ KOPYALAMA

CONTROL + V → KOPYALANAN İFADEYİ YAPIŞTIRMA

CONTROL + X → SEÇİLİ İFADEYİ KESME

CONTROL + D → DOSYADAKİ YER ALAN AYNI İFADELERİ SEÇME

CONTROL + N → YENİ DOSYA

CONTROL + ENTER → AL SATIRA GEÇİP YENİ SATIR AÇMA

CONTROL + SHIFT + P → AYARLAR

CONTROL + SHIFT + YÖN →BİR İFADEYİ SEÇME

CONTROL + ALT + YUKARI VEYA AŞAĞI →MULTICURSOR

ESC →MULTCURSORDEN ÇIKIŞ

ALT + TAB → PC DE PERNCERELER ARASI GEÇİŞ YAPMA

ALT + YÖN → BİR SATIRI YUKARI AŞAĞI TAŞIMA

En Çok Kullanılan HTML Etiketleri Nelerdir

Yazıya başlamadan önce **HTML** dilinde bir etiket nasıl oluşturulur bunu göstermek istiyorum. Etiketler küçüktür ve büyüktür işaretlerinin arasına yazılarak başlar (*örn.* `<etiket>`) içeriği yazdıktan sonra etiketi kapatmanız gerekir. Etiketi kapatmak için küçüktür, slash, etiket, büyüktür şeklinde yazılmalı (*örn.* `</etiket>`) Bütün bir örnek yapmak gerekirse (*örn.* `<etiket>` İçerik Yazılacak Alan `</etiket>`) **Not:** Bazı etiketler kapanmaya ihtiyaç duymazlar (*örn.* `<br>`, `<hr>`, `<meta>` vs.)

HTML Etiketi

HTML etiketi, dosya içeriğinin HTML dilinde yazıldığını tarayıcımıza bildirir. Bu etiket, kendi altında mutlaka `<head>` ve `<body>` etiketlerini barındırmalıdır. Eğer bir HTML dökümanı oluşturmak isterseniz ilk yapacağınız iş bir `<html>` etiketi oluşturmaktır. Ardından HTML etiketi altına `<head>` ve `<body>` etiketlerini yazmalısınız.

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

HEAD Etiketi

<head> etiketi, site ziyaretçileri tarafından görülmesi gerekmeyen kodları içerir. Bu etiket altına yazılan kodlar genellikle arama motorları ve örümcekler (Crawler veya Spider diye geçer) içindir. Head etiketi altında bütün etiketleri kullanabilmeniz mümkün değil. Kullanabileceğiniz etiketler;

- <title> (Bu etiketi kullanmak şarttır)
- <meta>
- <style>
- <script>
- <noscript>
- <link>
- <base>

```
<html>
  <HEAD>
    <title> Sekmede Görülecek İsim </title>
    <meta name="Keywords" content="HTML,Kodluyoruz">
  </HEAD>

  <BODY></BODY>

</html>
```

BODY Etiketi

Web sayfamızda görmek istediğimiz bütün içerikleri <body> etiketi altına yazıyoruz. Anlatacağım diğer etiketleri <body> etiketi içerisine yazacağız.

```
<html>
  <HEAD>
    <title> Sekmede Görülecek İsim </title>
    <meta name="Keywords" content="HTML,Kodluyoruz">
  </HEAD>

  <BODY>
    Site İçeriği
  </BODY>
</html>
```

Not:

Şu ana kadar oluşturduğumuz yapıyı idelerde kısayollar ile hızlıca oluşturabiliyoruz. Ben Visual Studio Code (VSC) kullanıyorum. VSC üzerinde "**! + Enter**" yazarak aşağıdaki yapıyı hızlıca oluşturabilirsiniz. <!DOCTYPE html> : Dökümanımızın **HTML** dilinde olduğunu tarayıcımıza bildiren talimattır. <html lang="en"> : Site içeriğinin dilini belirten etiket, "**en**" yerine "**tr**" yazabilirsiniz.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
</html>
```

H etiketleri başlık etiketleridir. Büyükten küçüğe sırasıyla

```
<h1>
<h2>
<h3>
<h4>
<h5>
<h6>
```

şeklindedir.

Not: HTML otomatik olarak **Başlık** etiketlerinin öncesine ve sonrasına satır atlatır.

P Etiketi

<p> etiketi paragraf etiketidir. Sayfa içerisinde oluşturacağımız metinleri <p> etiketi ile oluştururuz. Aşağıdaki örnekte hem başlık etiketi hem de paragrafa etiketini kullandım.

Not: HTML otomatik olarak **Paragraf** etiketinin öncesine ve sonrasına satır atlatır.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h2>Kodluyoruz</h2>
    <p>HTML Etiketleri</p>
</body>
</html>
```

BR Etiketi

 etiketi satır atlatma etiketidir ve kapatmaya ihtiyaç duymayan etiketlerden biridir. Atlatmak istediğiniz satır sayısı kadar
 etiketi kullanabilirsiniz. NOT: BR etiketinin farklı kullanımlarını görebilirsiniz. *örn.*(
,
,
) Hepsi aynı işlevi yerine getirir.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h2>Kodluyoruz</h2> <br>
    <p>HTML <br> Etiketleri</p>
</body>
</html>
```

A Etiketi <a> etiketinin en önemli özelliği **href** özelliğidir. Bu etiket ile sayfaları linkleyebiliriz. Etiket içerisine yazılan içerik sayfa üzerinde gösterilecek içeriktir. href içine yazılan ise tıklandığında gideceği URL'dir.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <a href="https://www.kodluyoruz.org">Kodluyoruz</a>
</body>
</html>
```

UL - OL - Li Etiketi

 ve <ol> etiketleri liste oluşturma etiketleridir. Listeyi oluşturduktan sonra içeriğini oluşturmak için <li> etiketini kullanıyoruz.

 = "**unordered list**" sırasız liste anlamına geliyor. = "**ordered list**" sıralı liste anlamına geliyor.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <ul>
        <li>HTML</li>
        <li>CSS</li>
        <li>JavaScript</li>
    </ul>
```

```
    <ol>
        <li>HTML</li>
        <li>CSS</li>
        <li>JavaScript</li>
    </ol>

</body>
</html>
## HR Etiketi

`<hr>` etiketi konusal bir geçişi temsil eder. Yazı yazarken yeni bir paragrafa başlamaya benzetebiliriz. Varsayılan olarak sayfaya yatay bir çizgi ekler ama bu özelliği değiştirilebilir. Bu etiket kapatılmaya ihtiyaç duymaz.

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Document</title>
  </head>
  <body>
    <p>
      HTML yani Hiper Metin İşaretleme Dili, web sayfalarında gördüğümüz iskelet
      yapısını oluşturmak için kullanılan metin işaretleme dilidir.
    </p>
    <hr>

    <p>
      CSS, HTML elementlerinin çeşitli medya araçlarında nasıl ekrana
      yansıtılacağını tanımlayan bir dildir. Tek seferde birden çok sayfa için
      kurallar belirtebilir.
    </p>

    <hr>
    <p>
      JavaScript, HTML ve web için bir programlama dilidir. Dinamil olarak HTML
      içeriğini ve özelliklerini, CSS'i değiştirebilme gibi işlemleri
      gerçekleştirebilir.
    </p>
  </body>
</html>
```
```

STRONG ve B Etiketi

 etiketi bir metinin arama motorlarına önemli olduğunu bildirmek için kullanılır. Kullanıldığı zaman metini kalın yapar. Eğer sadece metini kalınlaştırmak isterseniz <b> etiketini kullanabilirsiniz.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h2><strong> Kodluyoruz </strong></h2>
    <p><b> HTML </b> Etiketleri </p>
</body>
</html>
```

Script Etiketi <script> etiketi JavaScript kodlarını HTML içerisine yazabilmemizi sağlar.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        document.write("Kodluyoruz")
    </script>
</body>
</html>
```

Button Etiketi <button> etiketini buton oluşturmak için kullanırız. Buton üzerine yazmak istediğiniz içeriği etiketin içine yazmanız yeterlidir.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <button> Buton </button>
</body>
</html>
```

img Etiketi

Resim eklemek için`<img>` etiketini kullanıyoruz. `<img src=”resim.jpg” alt=”açıklama yazısı” />` **src=""** kısmına eklemek istediğimiz görselin yolunu yani kaynağını yazmalıyız. Eğer görselimiz ve HTML dosyamız aynı klasörde ise görselin adını ve uzantısını yazmamız yeterlidir. **alt=""** kısmına görselin açıklamasını yazıyoruz fakat isterseniz boş bırakabilirsiniz. Bu etiket kapanmaya ihtiyaç duymaz.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <img src=”görseller/kodluyoruz.jpg” alt=”Kodluyoruz Bootcamp” />
</body>
</html>
```

iframe Etiketi

Belge içinde belge gösterebilmemizi sağlayan etikettir. Genelde başka bir sitedeki belgeyi kendi sayfamızda göstermek için kullanırız. *örn:* Youtube'dan bir videoyu sayfamızda göstermek istersek `<iframe>` kodlarını sayfamıza eklememiz yeterli.(video üzerinde sağ tıklayıp yerleştirme kodunu kopyala diyerek iframe kodunu kopyalayabiliriz.)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <iframe width="1519" height="581" src="https://www.youtube.com/embed/BHPYQHnD_QA"
frameborder="0"
        allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope;
picture-in-picture"
        allowfullscreen></iframe>
</body>
</html>
```

Yorum Satırı

HTML dilinde yorum satırı`<!--` ile başlar `-->` ile biter.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <!-- Örnek Yorum Satırı -->
    <!--
        2. Örnek Yorum Satırı
    -->
</body>
</html>
```

Kaynaklar [w3schools]

En çok kullanılan html etiketleri

Bozmaktan düzenlemekten çekinme! Yazıları, linkleri değiştirebilir yenilerini ekleyebilirsin!

Başlık Etiketleri Kullanımı

Efendim öncelikle HTML'in ne olduğunu ve en yaygın etiketleri öğrendiniz. Şimdi geldik yavaş yavaş biraz daha bu güzel betik dilinde ilerlemeye. Bu yazıda `<head></head>` etiketi arasına yazılan etiketlerden bahsedeceğiz.

Bildiğiniz üzere balık baştan kokar derler. Başlık etiketlerimiz de benzer şekilde bize sayfamızla ilgili ipuçları verir ve sayfamız için ihtiyacımız olacak dış kaynakları yüklememize yardımcı olur. Baklavayı ortasından yemeyi sevenler için biraz daha detay vermek gerekirse, hem sayfamızın düzgün ve verimli çalışması için hem de bu sayfanın tanınmasını bilinmesini hatta arama motorları tarafından fark edilmesini sağlamak için bu etiketlere ihtiyacımız var.

Öyleyse sırasıyla etiketlerimize başlayalım ve html sayfalarımızı nasıl daha bilinir ve gelişmiş hale getirebiliriz öğrenelim.

Title Etiketi

Bu etiketimiz html dökümanlarında türkçe anlamı olan "başlık" görevini üstlenir. Bu başlık birkaç farklı yerde görülebilir. Bizim en sık karşılaştığımız şekli ise browser sekmelerinin isimleridir. Örnek olarak :

```
<title> Kodluyoruzla Web Öğreniyorum </title>
```

Gibi bir başlık belirlediğimizde sekme isminde Kodluyoruzla Web Öğreniyorum yazdığını görürüz.

Bu etiket arasına yazdıklarımız aynı zamanda sayfayı favorilere eklerken de karşımıza çıkar.

Ayrıca arama motorları (Google, Yahoo, Bing ...) sayfamızın bu kısmına bakarak sitemizi listeler. Bu yüzden de oluşturduğumuz mükemmel web sayfalarının insanlara ulaşması için bu alanda yazdıklarımızın açıklayıcı, dikkat çekici ve 50-60 karakteri geçmeyecek şekilde (Arama motorları genelde başlığın ilk 50-60 karakterini gösterir) yazmamız gerekir.

Style ve Script Etiketleri

HTML dökümanları oluştururken en çok kullanacağımız etiketler bunlardır. Bu etiketler yalnızca `<head></head>` etiketleri arasında bulunmaz ancak biz burada bulunabilecek bazı özel türleri ve bu tür etiketlerin özelliklerini (attribute) inceleyeceğiz.

Style Etiketleri

Öncelikle style etiketinden başlayalım. `<style></style>` etiketleri arasında sayfamızı güzelleştiren, renklendiren belli özellikler tanımlayabiliyoruz. Bu kısımlarda, bir html dökümanında hangi alanın nerede ve nasıl görünmesi

gerektiğini tasarlayabiliriz. Belli kuralları olan bu belirteçlere CSS diyoruz. Sayfa tasarımlarının bulunduğu bir ön döküman veya bir taslak gibi düşünebiliriz. Daha CSS ile daha detaylı bilgi için şurayı inceleyebilirsiniz : https://www.w3schools.com/css/default.asp

Burada dikkat etmemiz gereken bir konu var. HTML dökümanı işlenirken ve görüntülenirken sayfa sırayla işlendiği için her zaman sırasıyla en altta kalan stil belirlemeleri baskın gelecektir.

Bu etiketimizin global özelliklere(attribute) ek olarak alabildiği iki farklı özellik vardır. Bunlar media ve type. Çok yaygın olmadığı için kullanım detaylarına girmeyeceğim ama alabildikleri özelliklere şuralardan bakabilirsiniz : https://www.w3schools.com/tags/att_style_media.asp

Script Etiketleri

Geldik web dökümanlarının ön yüzlerini "sihirli" hale getiren etikete. Bu etiketle web sayfalarının, browser yardımıyla çalıştırabildiği kodlar yazabiliriz. Sayfamızı canlandırabilir, hareketlendirebilir her alanda değişiklik yapabiliriz. Sayfamız üzerinde hayal gücümüzle sınırlı her değişikliği yapabilmemizi sağlayan kodlar bu etiketler arasında bulunur.

`<script></script>` etiketleri ileride öğreneceğiniz sihirli bir dünyaya açılan kapıdır. Normalde HTML dökümanlarının bir programlama dili ile yazılmadığını biliyoruz. Çünkü HTML ve CSS bir programlama dili değildir. Ancak`<script></script>` tagları arasına girdiğimizde işler değişmeye başlıyor. Artık bir programlama dili olan JavaScript'in dünyasına girmiş oluyoruz. HTML dökümanlarının stilleri yerleşimleri hatta bütün dökümanın kendisini Javascript yardımıyla değiştirebilir, farklı işlemler yapabilir, farklı sayfalarla veya arka planda bir veri tabanıyla haberleşebilir oradan aldığımız bilgilerle dökümanımızı güncelleyebiliriz. Ayrıca HTML dökümanımıza yeni elemanlar ekleyip çıkartabiliriz. Dökümanlarımız üzerinde hareketli animasyonlar çalıştırabilir, her türlü değişikliği yapabiliriz.

Bu yüzden bu etiketlerin bizim gücümüzü artıran sihirli bir dünyaya açıldığını söyleyebiliriz. Şimdi bunun özelliklerine bakalım.

Script Tag Özellikleri

Öncelikle bu etiketlerin bize sayfa üzerinde büyük bir güç verdiğinden bahsetmiştik. Ancak Spider Man Peter Parker'ın rahmetli amcasının da dediği gibi "with great power comes great responsibility" yani büyük güç büyük sorumluluk gerektirir. Bu yüzden bu tagın özelliklerinden bahsederken sayfanın işlenişi, browser tarafında nasıl işlendiğini ve yanlış bir kullanımın nelere sebep olabileceğini iyi anlamamız gerekiyor. Eğer script etiketini kullanırken herhangi bir özellik eklemezsek browser sırası geldiğinde doğrudan işlenir. Ve bu kısım işlenmeden sayfa yüklenmeye devam etmez. Bu noktada da *async* özelliğimiz devreye giriyor. Eğer sayfanın yüklenmeye devam ederken eşzamanlı olarak bu etiketlerle belirlediğimiz scriptlerin de yüklenmesini ve hazırlanmasını istiyorsak, yani bu kısmın asenkron çalışmasını istiyorsak etiketimize bu özelliği ekliyoruz. Herhangi bir değer girmemize gerek yok şu şekilde kullanabiliriz :

```
<script src="myJavascript.js" async></script>
```

Eğer bu etiketin sayfa yüklendikten sonra yüklenip çalıştırılmasını istiyorsak o zaman async özelliğinin yanına *defer* özelliğini de eklememiz gerekiyor. Ancak bu iki özellik de yalnızca sayfa harici kaynaktan yani bu HTML içinde yazmadığımız javascripti yüklerken kullanabileceğimiz özellikler. Buna da dikkat etmemiz gerekiyor.

Bu etiketin sihirli dünyasından bahsetmiştik. Ancak bahsetmediğimiz nokta her sihirli dünyada olduğu gibi burada da kötü büyücülerin, kötü insanların olduğudur. Bu yüzden sayfamız üzerinde bu kadar güce sahip bir alanın doğru şekilde korunması ve kullanılması hayati önem taşıyor. İşte bu amaçla da karşımıza iki özellik çıkıyor.

Bu özelliklerden birincisi crossorigin. Browserlar, istek sahteciliği gibi güvenlik sorunlarıyla aktif şekilde mücadele etmeye çalışıyor. Bu yüzden bir kaynaktaki dökümanın bir diğer kaynaktaki (farklı domain) dökümanlara erişmesinde biraz hassas davranıyorlar. Bu konuya cross origin resource sharing deniyor kısaca CORS diyebiliriz. İşte bu etiketimiz de farklı kaynaklardaki, farklı domainlerdeki scriptleri yüklememiz için bize yardımcı oluyor. Eğer bir kaynaktan(aynı domain dahil) bir şey yüklemek için belli bilgileri(Çerezlerimiz olabilir, HTTP basit giriş bilgileri olabilir) göndermemiz gerekiyorsa bu özelliğin değerini crossorigin = "use-credentials" olarak belirlemeliyiz. Eğer anonim şekilde erişmemiz gerekiyorsa yani herhangi bir bilgiye ihtiyaç yoksa crossorigin="anonymous" olarak kullanacağız.

Bir diğer özelliğimiz ise integrity özelliğidir. Integrity türkçeye bütünlük, doğruluk, dürüstlük şeklinde çevirilebilir. Webin gelişmesiyle birlikte bir HTML sayfasına yüklenen kaynaklar çoğaldı. Özellikle tekrar eden ihtiyaçlar için zaman geçtikçe en verimli çözümler üretilmeye ve kullanılmaya başlandı. Bu çözümlerin kullanılması yaygınlaştıkça belli riskler de ortaya çıkmaya başladı. Örnek olarak HTML sayfamıza ekleyeceğimiz bir dış script bir güvenlik açığıyla karşı karşıya kaldığında o scripti kullanan bütün sayfalar aynı anda etkilenmiş olacak. Bunun bir örneği yakın zamanda birçok firmayı etkileyen Gigya servisinde olmuştu. Bu servisi kullanan sitelerin sayfalarına ekledikleri harici bir gigya servisi bir saldırıya maruz kalmıştı ve o servisi kullanan binlerce sitede bu açıktan etkilenmişti. İşte CDN dediğimiz bu gibi hazır scriptleri eklediğimiz durumlarda bir doğrulama yöntemine ihtiyaç duyuyoruz. Yani browser bir şekilde, sayfamızı etkileyecek kodların bizim istediğimiz eklediğimiz kodlar olduğunu doğrulaması gerekiyor. Bu noktada da integrity özelliği devreye giriyor. Sayfamızda kullanacağımız hazır kodların bir imzasını bu özelliğe değer olarak ekliyoruz. Bu imza doğrudan kodun kendisinden oluşturulur ve belli bir karakter uzunluğundadır. Ayrıca kodda bir harf bile değişecek olsa imza tutmayacaktır. Bu sayede eğer kodda zararlı/zararsız herhangi bir değişiklik olursa browser imzalar uyuşmayacağı için kodları sayfamıza yüklemeyecektir.

Örnek olarak şu şekilde kullanabiliriz :

```
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"
integrity="sha384-0mSbJDEHialfmuBBQP6A4Qrprq5OVfW37PRR3j5ELqxss1yVqOtnepnHVP9aJ7xS"
crossorigin="anonymous"></script>
```

Bir diğer etiketimiz de refreferrerpolicy. Bu etiket de scripti yükleyeceğimiz zaman, alacağımız kaynağa atacağımız verileri eklemek için kullınılır. Detaylı kullanımına buradan bakabilirsiniz :https://www.w3schools.com/tags/att_script_referrepolicy.asp Bu da crossorigin gibi kaynak paylaşımı maksadıyla kullanılan özelliklerdendir.

HTML sayfamızı oluştururken sayfa içerisindeki kod ne kadar uzun olursa okunması, yazılması ve incelenmesi o kadar zor olur. Bu yüzden kodları farklı sayfalara bölüp kullanmak hem daha kullanışlı hem de daha verimli olur. İşte bu amaçla farklı sayfalardaki scriptleri yükleyebilmek için de script etiketini kullanabiliriz. Bu amaçla script etiketinin *src* özelliğini kullanırız. Bu özellikle hem kendi dosya sistemimizde hem de internet üzerinde herhangi bir adreste bulunan kodları kendi sayfamıza ekleyebiliriz.

Örnek olarak kendi dosya sistemimizde, html dökümanımızla aynı dizinde bulunan bir script dosyasını çağırmak için:

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

Veya bir web sayfasındaki başka bir scripti çağırmak için :

```
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"
integrity="sha384-0mSbJDEHialfmuBBQP6A4Qrprq5OVfW37PRR3j5ELqxss1yVqOtnepnHVP9aJ7xS"
crossorigin="anonymous"></script>
```

Son olarak da bu etiketle kullanabileceğimiz type özelliğimiz, yüklenecek dosyanın içeriğinin, browser tarafından nasıl yorumlanması gerektiğini belirtir. Örnek olarak javascript dosyası yüklemek için yüklenecek kodların javascript kodunu belirtmek için şöyle yazabiliriz :

```
<script type="application/javascript">
document.getElementById("someTestDiv").innerHTML = "This code runs as js";
</script>
```

Ya da Ecmascript için :

```
<script type="application/ecmascript">
document.getElementById("someTestDiv").innerHTML = "This code runs as js";
</script>
```

Link Etiketi

Script etiketini anlatırken bir HTML dökümanında yüklenecek kodların bölünmesinin bir çok kolaylık sağlayacağından bahsetmiştik. İşte `<link></link>` etiketi de script etiketinin src özelliği ile kullanılması gibi link etiketini de farklı kaynaklardan farklı dosyaları HTML dökümanımıza dahil etmek için kullanabiliriz. Bu etiket dökümanımızda bulunmayan dış kaynaklarla dökümanımız arasındaki ilişki kurmak için kullanılır.

Bu etikette de crossorigin özelliği mevcuttur.

Bu etikette ilişki kurmak istediğimiz dış kaynağı href özelliği ile veriyoruz. Bu özelliğin açılımı Hypertext REFerence şeklindedir. Örnek olarak bir CSS sayfasını HTML dökümanımız ile ilişkilendirmek için şu kodu kullanabiliriz :

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

href özelliğini hem dış kaynaklardan hem de dökümanımızın bulunduğu dosyadan ilişkilendirme yapmak için kullanabiliriz.

```
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
integrity="sha384-JcKb8q3iqJ61gNV9KGb8thSsNjpSL0n8PARn9HuZOnIxN0hoP+VmmDGMN5t9UJ0Z"
crossorigin="anonymous">
```

Bu etiketin önemli özelliklerinden birisi de *rel* özelliğidir. Bu özellik dış kaynaktaki dosyayı kendi HTML dökümanımıza ne şekilde ilişkilendirmek istediğimizi belirttiğimiz kısımdır. En çok kullanılan değerleri "stylesheet" ve "icon" dur. Stylesheet eklemek istediğimiz dosyanın bir stil dosyası olduğunu söyler. Böylece browser oradaki komutları HTML'imizi şekillendirmede ve değiştirmede kullanır.

Yukarıda HTML dökümanımızın stil ve script dosyalarını ayrı ayrı yazmanın faydalarından bahsetmiştik. Script etiketi ile bu javascript dosyalarını ekleyebiliyorduk ancak CSS yani stil dosyalarını eklemenin yöntemini öğrenmemiştik. İşte bu link etiketi ve rel özelliği yardımıyla istediğimiz stil dosyasını da `<head></head>` etiketlerinin arasında ekleyerek web sayfamızı daha renkli ve eğlenceli hale getirebiliriz.

`<title></title>` etiketi yardımıyla sekmelerde görünecek, HTML dökümanımızın başlık ismini belirlemiştik. Ancak şu an bu yazının olduğu sekmeye bakarsanız başlığın yanında bir de küçük bir logo/ikon göreceksiniz. İşte bu ikonlar eklemek için de rel = "icon" kullanıyoruz. Bulunduğumuz dizinde **.ico** uzantılı bir **favicon.ico** dosyamız varsa bu dosyayı sekmelerde ismin yanında ikon olarak göstermek üzere şöyle bir kod kullanabiliriz :

```
<link rel="icon" href="/favicon.ico" type="image/x-icon">
```

Son olarak link etiketi de type özelliği kullanır. Bu özellikle de ilişkilendirdiğimiz dosyanın tipini vermiş oluyoruz. Yaygın kullanılan değerleri stil dosyaları için `type = "text/css"` şeklinde, ikonlar için de `type="image/x-icon"` şeklindedir.

Link etiketi global özellikleri de destekler. Diğer detaylı özellikleri için de buraya bakabilirsiniz : https://www.w3schools.com/tags/tag_link.asp

Meta Etiketi

Tanıtacağımız son etiket ise meta etiketi. Meta veriler bilgisayar bilimleri dünyasında genellikle verinin verisi anlamında kullanılır. Yani bir veriyle ilgili bilgiler meta bilgiler olarak tanımlanır. İşte HTML dökümanımızla ilgili verilerin olduğu etiketler de meta etiketleridir. Burada vereceğimiz bilgiler sitemizi arama motorlarına, sosyal medyaya ve diğer sitelere tanıtmak ve dökümanımızla ilgili bilgiler vermek için kullanılacak veriler olacak.

Sadece 4 farklı özelliği olan `<meta></meta>` etiketi dökümanımızla ilgili birçok bilgiyi barındıran farklı kullanım şekillerine sahip. Bu kullanımlar sayfamızla ilgili önemli bilgiler içerdiği için ayrı ayrı inceleyeceğiz.

Dünyada farklı farklı diller ve farklı alfabeler mevcut. Örnek olarak latin alfabesi, arap alfabesi, çin alfabesi, elf alfabesi vs vs. Ancak hepsinde farklı karakterler olduğu için browserin bu karakterli görüntüleyebilmek için doğru şekilde çözümlemesi gerekir. İşte HTML dökümanımızdaki bu karakterlerin çözümlenme biçimlerini belirttiğimiz `<meta>` etiketi özelliğimiz *charset* özelliğidir. Bizim latin alfabesi için verilen charset kodu UTF-8 dir.

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

Bir diğer önemli özelliğimiz ise http-equiv'dir. Browserlar farklı sunuculara istek atarlarken belli bilgileri karşı tarafa gönderirler. İşte bu isteklerin arasında isteğin detaylarıyla ve yöntemiyle ilgili bilgilerin olduğu header'lar bulunur. Biz de dökümanımızda o dökümana ulaşan birisinin browser'inde header alanında bir bilgi tutmak istiyorsak bu meta etiketi özelliğini kullanabiliriz. Örnek olarak charset ile belirttiğimiz özellik HTML5 ile gelmiştir. Daha önceki versiyonlarda ise şu şekilde bir kullanım vardır :

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

Ayrıca refresh başlığını(header) bu meta yardımıyla belirleyerek sayfamızın belli sürede bir yenilenmesini veya belli bir süre sonra başka bir sayfaya yönlendirilmesini sağlayabiliriz.

```
<meta http-equiv="refresh" content="10;URL=kodluyoruz.html">
```
Yukarıdaki kodda sayfa yüklendikten 10 saniye sonra URL ile verdiğimiz değer olan kodluyoruz.html'ye yönlendirilecek.

Burada kullandığımız diğer etiket de *content* etiketidir. Bu da meta olarak verdiğimiz bilgilerin içeriğini tanımlamamızı sağlar.

Son özelliğimiz de *name* özelliğimizdir. Bu da meta bilgi olarak vereceğimiz bilginin tanımlayıcısıdır diyebiliriz. Örnek olarak sayfamızda en çok geçen harfin ne olduğunu belirteceğimiz bir meta bilgisi yazmak isteyelim:

```
<meta name="enCokGecenHarf" content="a">
```
Bu şekilde istediğimiz meta bilgiyi sayfamızın başlık etiketleri arasında kullanabiliriz.

Buraya kadar meta etiketini öğrendik peki sosyal medyalar ve arama motorları sitemizle ilgili bilgileri nasıl bu meta verilerden alıyor? Bu sorunun cevabı da yaygın kullanılan meta etiketleri kalıplarında bulunuyor. Örnek olarak github'da sağ tıklayıp sayfa kaynağını görüntüle dediğimizde karşımıza bir çok meta etiketi çıkıyor :

```
<meta property="og:url" content="https://github.com">
<meta property="og:site_name" content="GitHub">
<meta property="og:title" content="Build software better, together">
<meta property="og:description" content="GitHub is where people build software. More
than 50 million people use GitHub to discover, fork, and contribute to over 100
million projects.">
<meta property="og:image"
content="https://github.githubassets.com/images/modules/open_graph/github-logo.png">
<meta property="og:image:type" content="image/png">
<meta property="og:image:width" content="1200">
<meta property="og:image:height" content="1200">
<meta property="og:image"
content="https://github.githubassets.com/images/modules/open_graph/github-mark.png">
<meta property="og:image:type" content="image/png">
<meta property="og:image:width" content="1200">
<meta property="og:image:height" content="620">
<meta property="og:image"
content="https://github.githubassets.com/images/modules/open_graph/github-
octocat.png">
<meta property="og:image:type" content="image/png">
<meta property="og:image:width" content="1200">
<meta property="og:image:height" content="620">

<meta property="twitter:site" content="github">
<meta property="twitter:creator" content="github">
<meta property="twitter:card" content="summary_large_image">
<meta property="twitter:title" content="GitHub">
<meta property="twitter:description" content="GitHub is where people build software.
More than 50 million people use GitHub to discover, fork, and contribute to over 100
million projects.">
<meta property="twitter:image:src"
content="https://github.githubassets.com/images/modules/open_graph/github-logo.png">
<meta property="twitter:image:width" content="1200">
<meta property="twitter:image:height" content="1200">
```
İşte burada kullanılan meta taglar, facebook ve twitter paylaşımlarımızda bir url paylaşmak istediğimizde karşımıza çıkıyor. Sosyal medyada bir url paylaşmak istediğimizde url'yi kopyaladıktan sonra bir kart görürüz. Bu kartta bir başlık bir resim ve kısa bir açıklama bulunur. İşte o bilgiler bu meta verilerden alınır.

Arama motorları bizi meta verilerinde verilen "keywords" lere göre sıralamaya alır :

```
<meta name="keywords" content="Kodluyoruz,programlama,web">
```
Ayrıca arama motorlarında sayfa başlığının altındaki açıklamalarda da yine bu meta verilerine öncelik verilir :

```
<meta name="description" content="Kodluyoruzla web öğrenmeye hazır mısınız?">
```
Bazı arama sonuçlarında yazar adı görürsünüz. Bu kısım da yine meta etiketinin marifetidir:

```
<meta name="author" content="Kodluyoruz">
```
Meta etiketiyle söyleyeceğimiz son şey de viewport konusu. Akıllı telefonlarla birlikte geliştiriciler artık farklı cihazlara, farklı cihaz ekranlarında düzgün görünen kodlar yazmaya çalışıyor. Her ekranın genişliği boyutları farklı olduğu için tek bir ekran türüne göre tasarım ve kodlama yapmak da bu sorunu çözmüyor. Bu yüzden farklı cihazlarda da iyi görünen siteler yapmak için temel görünen alanı, bu alanın genişliğini vs tanımlamamız gerekiyor. İşte viewport burada yardımımıza koşuyor.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```
Burada genişliğin cihaz genişliğinde olduğunu ve zoom oranının 1.0 olacağını tarayıcıya belirmiş oluyoruz. Böylece mobilde masaüstü görünümü gibi bir görünüm değil olması gerektiği gibi düzgün bir görüntü elde ediyoruz. Detaylı bilgi için şurayı inceleyebilirsiniz : https://fatihhayrioglu.com/meta-viewport-etiketi/

Şimdiye kadar başlık etiketlerini, başlık etiketlerinin özelliklerini ve genel kullanım alanlarını öğrenmiş olduk.

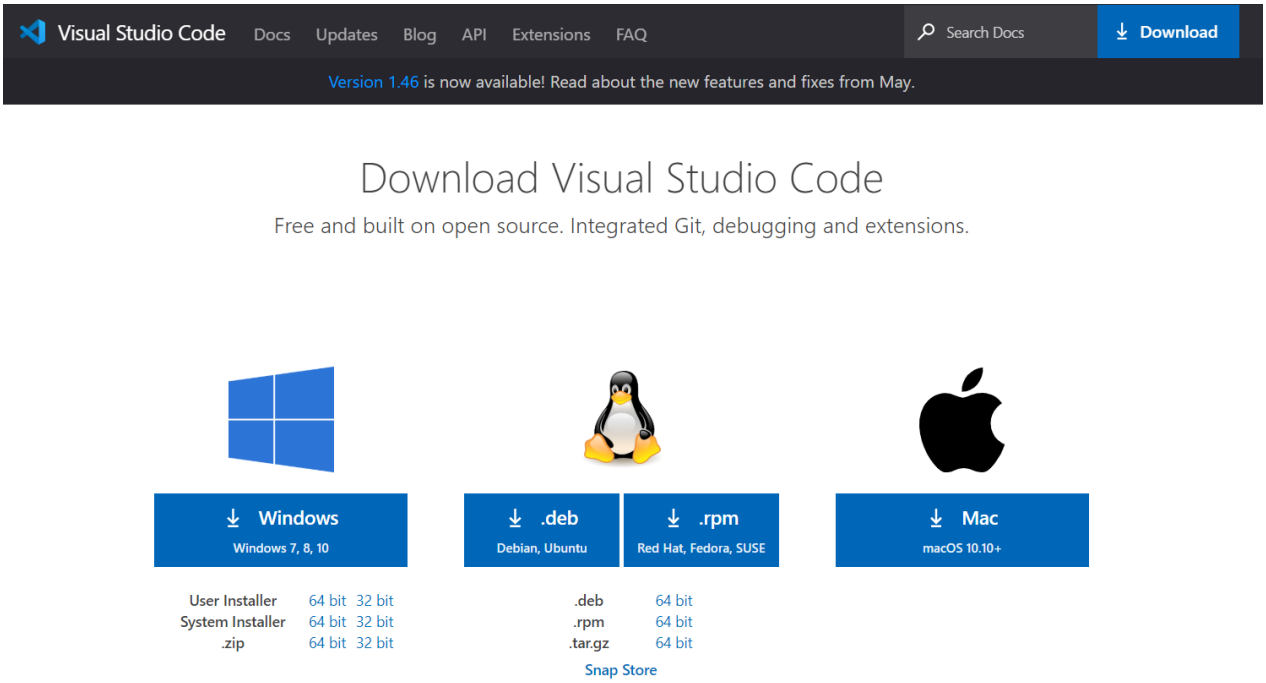
```
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
integrity="sha384-JcKb8q3iqJ61gNV9KGb8thSsNjpSL0n8PARn9HuZOnIxN0hoP+VmmDGMN5t9UJ0Z"
crossorigin="anonymous">
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"
integrity="sha384-B4gt1jrGC7Jh4AgTPSdUtOBvfO8shuf57BaghqFfPlYxofvL8/KUEfYiJOMMV+rV"
crossorigin="anonymous"></script>
```

2. Bu tip sayfayla ilgili veriler için `<meta>`etiketini kullandığımızdan bahsetmiştik. Ayrıca başlık için de `<title>` etiketini kullandığımızı söylemiştik. Şimdi bu etiketlerin doğru kullanımlarına bakalım :

```
<title>Kodluyoruz HTML</title>
<meta name="dominantColor" content="red">
<meta name="description" content="Kodluyoruz ile webe giris">
```
Kaynaklar

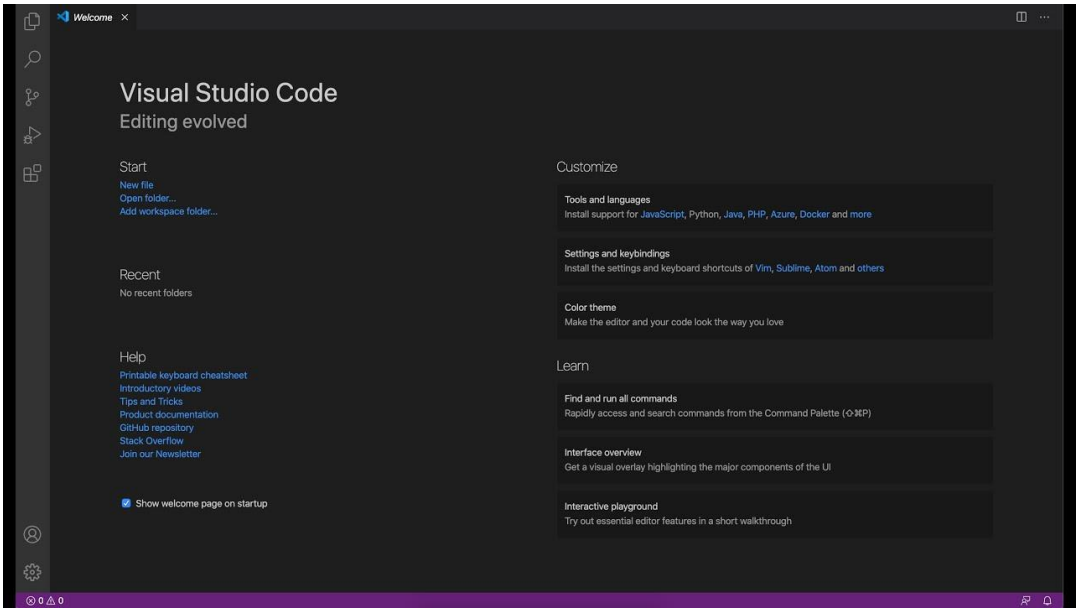
- https://clutch.co/seo-firms/resources/meta-tags-that-improve-seo
- https://www.geeksforgeeks.org/most-commonly-used-tags-in-html/
- https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML/The_head_metadata_in_HTML
- ttps://www.w3schools.com/tags/tag_script.asp
- https://www.w3schools.com/tags/tag_head.asp
- https://www.w3docs.com/learn-html/html-head-tag.html
- https://help.ahrefs.com/en/articles/2433739-what-is-meta-refresh-redirect-and-why-is-it-considered-a-critical-issue
- https://en.wikipedia.org/wiki/Meta_refresh
- http://www.yunusbassahan.com/blog/genel/arama-motorlari-ve-sosyal-medya-servisleri-icin-meta-tag-ler.html
- https://stackoverflow.com/questions/6535405/what-is-the-attribute-property-ogtitle-inside-meta-tag
- https://www.w3schools.com/tags/tag_meta.asp
- https://gist.github.com/lancejpollard/1978404
- https://www.w3schools.com/tags/tag_link.asp
- https://www.w3schools.com/tags/att_link_rel.asp
- https://www.w3schools.com/tags/att_link_type.asp
- https://developer.mozilla.org/en-US/docs/Web/HTML/Element/link
- https://developer.mozilla.org/en-US/docs/Web/HTML/Link_types
- https://www.w3schools.com/tags/att_meta_charset.asp
- https://www.w3schools.com/tags/att_meta_name.asp
- https://stackoverflow.com/questions/2359866/html-set-image-on-browser-tab
- https://stackoverflow.com/questions/34429024/what-is-the-purpose-of-the-integrity-attribute-in-html
- https://www.w3schools.com/tags/att_meta_http_equiv.asp

`Visual Studio Code Nedir?` Visual Studio Code (VS Code) bütün işletim sistemlerinde çalışabilen ve yapacağımız yazılımlar için kodlarımızı yazabileceğimiz bir text editörüdür. **`Visual Studio Code Yükleme`** VS Code programını ücretsiz bir şekilde bu adresten indirebilirsiniz. Bilgisayarınız işletim sistemi ve işlemci özelliklerine göre indireceğiniz VS Code programını bilgisayarınıza kurabilirsiniz.



`Visual Studio Code ile Dosyaların Oluşturulması`

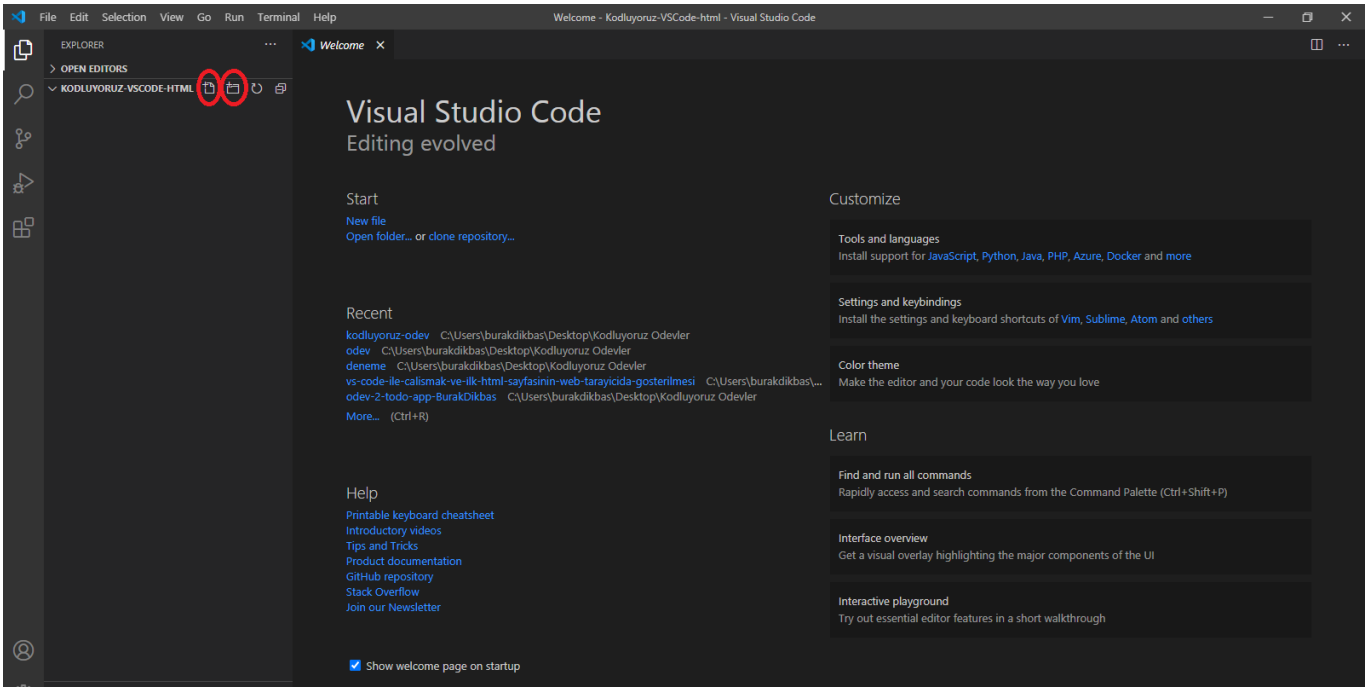
Öncelikle VS Code'da çalışabilmemiz için bilgisayarımızda çalışacağımız klasörü oluşturmamız gerekmektedir. Örneğin masaüstünde çalışmak istiyorsak ve yazdığımız kodları burada tutmak istiyorsak ilk olarak masaüstünde bir klasör oluşturmamız gerekmektedir. Klasör oluşturma işlemini tamamladıktan sonra bilgisayarımıza yüklediğimiz VS Code programımızı çalıştırıyoruz ve karşımıza şu şekilde bir ekran çıkıyor;



Bu ekran üzerinde open folder'ı tıkladıktan sonra az önce oluşturduğumuz ve çalışmak istediğimiz klasörümüzü seçiyoruz veya oluşturduğumuz dosyayı sürükle bırak yöntemiyle de açabiliriz. Proje klasörümüz açıldıktan artık bu

ekranda proje geliştirme işlemlerimizi yapabiliriz. Burada dikkat etmemiz gereken en önemli konu oluşturduğumuz dosya ismiyle VS Code'a gelen klasör adının ismi aynı olmalıdır. Eğer bir farklılık var ise klasör seçerken bir yanlışlık yapmışız demektir.

Artık proje oluşturacağımız ana klasörümüzü oluşturduk bundan sonra oluşturacağımız klasörleri veya dosyaları ana klasörümüzün yanında bulunan ikonlardan yapabiliriz.

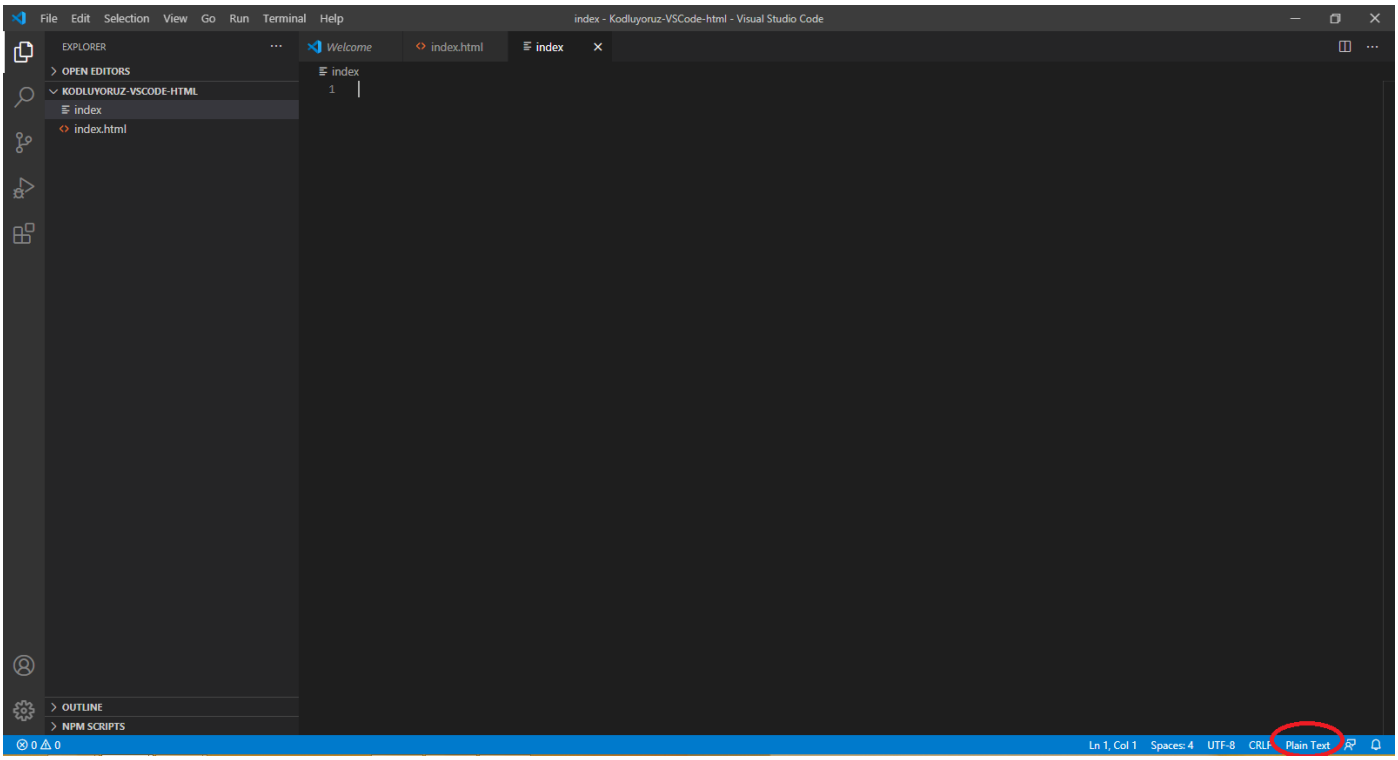


Solda bulunan ikondan dosya, sağda bulunan ikondan ise klasör oluşurabiliriz. Burada açacağımız dosya ve klasörlerimiz oluşturduğumuz ana klasörün altına otamatik olarak gelecektir. Bu alanda iç içe klasörler oluşturup içerisindeki klasörler veya dosyaları sürekle bırak yöntemi ile istediğimiz düzeni oluşturabiliriz.

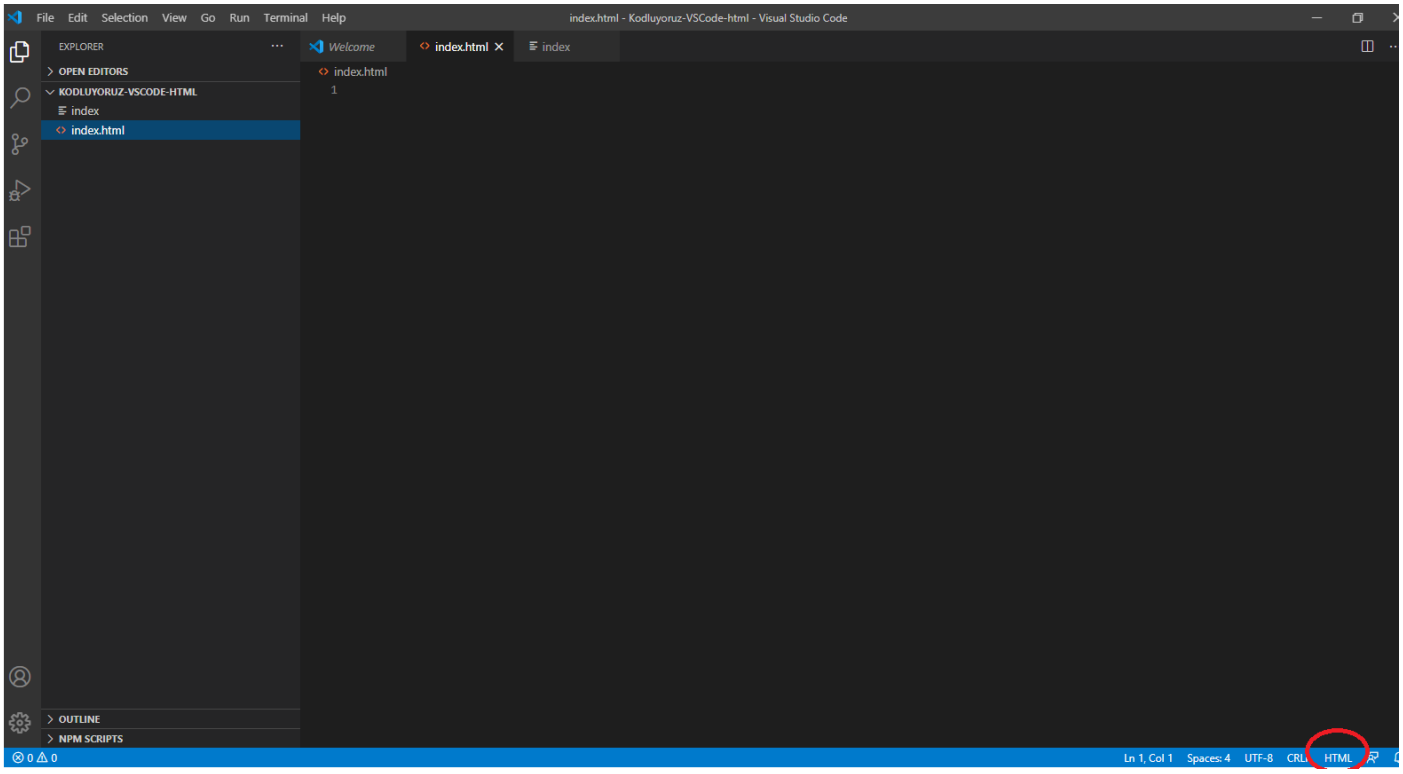
Visual Studio Code ile HTML Dosyasının Oluşturulması ve Web Browserda Gösterilmesi

Klasörümüzü oluşturduğumuza göre artık ilk html klasörümüzü oluşturabiliriz. Bunu oluşturmak için ana klasörümüzün yanında bulunan "new file" ikonuna basıyoruz. Açılan dosyada bizim dosyaya bir isim vermemiz beklenmektedir. Yazacağımız bu isim text editörümüzün html etiketlerini tanıyabilmesi için oldukça önem taşımaktadır. Eğer dosyamızın uzantısını .html şeklinde yapmazsak text editörümüz html etiketlerini tanımayacaktır.

.html uzantılı yapmadığımız dosya



.html uzantılı yaptığımız dosya

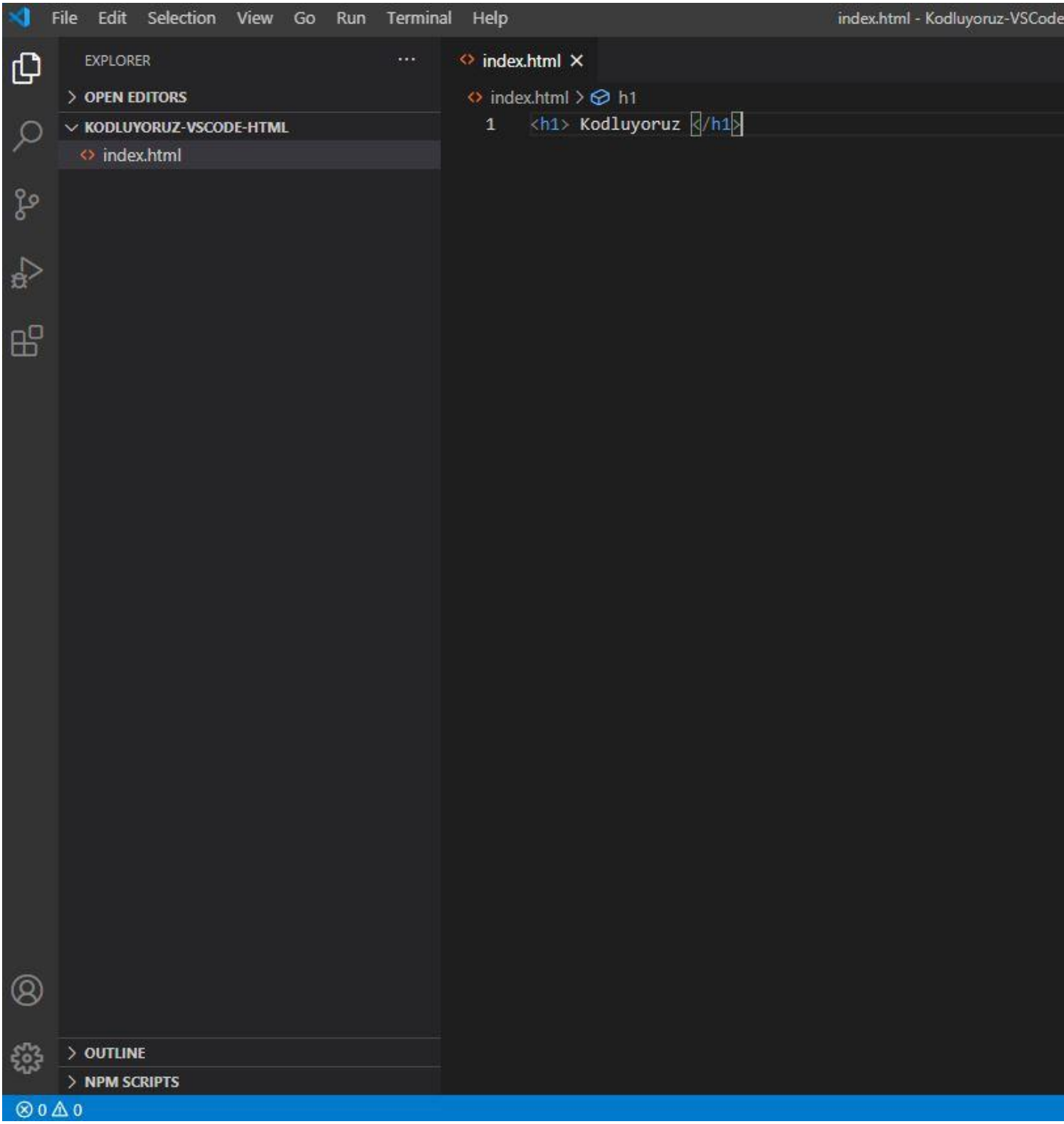


İkisine baktığımızda birinde sağ altta Plain Text yazarken diğerinde html etiketlerimizi anladığını gösteren HTML ibaresi yer almaktadır.

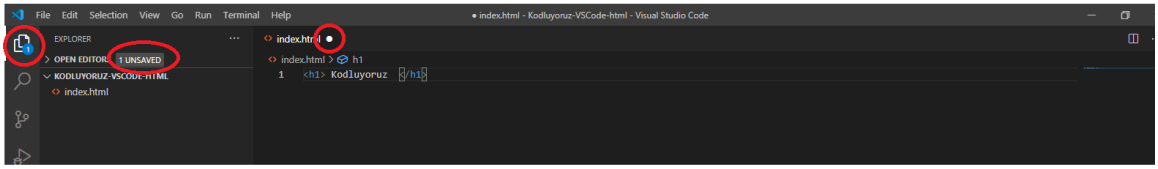
Doğru olarak yazdığımız index.html dosyamızda ilk html etiketimizi oluşturalım. Başlık etiketi için kullanılan h1 etiketi ile alıştırmamızı yapalım.

```
<h1> Buraya İstediğimiz Başlığı Yazabiliriz </h1>
```

h1 etiketleri arasına istediğimiz bir başlığı yazabiliriz. Aşağıda verilen örnekte h1 etiketlerimiz arasına "Kodluyoruz" yazılmıştır.

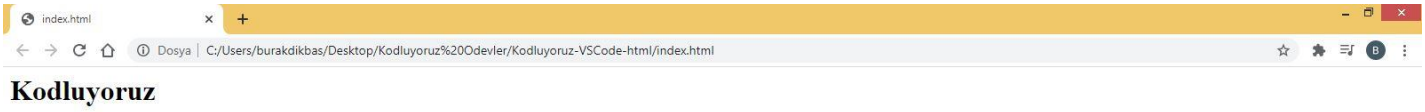


Oluşturduğumuz dosyamızı çalıştırmadan önce kaydetmemiz gerekmektedir. Eğer bir değişiklik yapmış ve dosyamızı kaydetmemişsek yazdığımız dosyanın üzerinde bir nokta ve en solda Explorer'da kaç dosyanın kayıtlı olmadığı gösteren bir ikon göreceğiz.



Oluşturduğumuz index dosyamızı File > Save ile veya CTRL+S ile kaydettikten sonra ilk html etiketimizi web browserda görüntüleyebilmek için ana klasörümüzün altında oluşturduğumuz index dosyamızı açıyoruz.

veeee ilk HTML sayfamız web tarayacımızda görülmekte!!!



Kaynaklar

- Kodluyoruz Frontend 101 Video Eğitimi - Hakan Yalçınkaya
- https://code.visualstudio.com/
- https://guides.github.com/features/mastering-markdown/

Açıklama Satırları Oluşturmak

Web sayfalarımızı hazırlarken açıklama satırlarını nasıl oluştururuz, hangi durumlarda kullanırız, amacı nedir bu açıklama satırlarının gibi konuları inceliyor olacağız.

Web sayfalarımızı hazırlarken bazı durumlarda açıklama satırlarına ihtiyaç duyarız. Eğer HTML'e başlamadan önce herhangi bir yazılım dili ile ilgilendiyseniz yorum/açıklama satırlarını kullanmışsınızdır. HTML tarafında açıklama satırlarının yazım stili diğer dillere göre daha farklı.

Açıklama satırı eklemeli miyim?

Evet, eklemelisin. Çünkü açıklama satırlarını kodunuzu 2 ay sonra tekrar gözden geçirmek için açtığınızda neyin nerede olduğunu anlamanız açısından fayda sağlayacaktır. Sadece kendimiz ile de bitmiyor. Projeyi farklı bir kişiye devrettiğinizde veya paylaştığımızda yazdığınız yorum satırlarının başka biri tarafından okunması karşı taraf için de fayda sağlayacaktır. Hatta yazdığımız kodların reçetesi veya ilacın üzerine yazılan kullanım talimatı gibi yorumlayabiliriz. :)

Açıklama satırı nasıl oluşturulur?

Açıklama satırını HTML kodumuzda `<!-` ile başlatıp `->` ile bitecek şekilde yazabiliyoruz. "ile başlatıp" yazan alana yorumlarımızı ekleyebiliyoruz.

Açıklama satırı web sayfasında görünür mü?

Hayır, kullanıcılar web sayfanızda açıklama satırlarını görmezler. Ayrıca açıklama satırları web tarayıcıları tarafından herhangi bir işleme tabi tutulmazlar. Fakat kullanıcı sayfanızda sağ tıklayıp kaynağı görüntüle dediğinde açıklama satırlarınızı görebilir.

Hadi örneklendirelim!

Aşağıdaki örnekte birden fazla HTML elementlerini tanımlayıp aralara açıklama satırları ekliyoruz. Böylelikle HTML kodumuz daha okunabilir bir hal alabiliyor. Hangi elementin nerede kapatıldığını veya açıklama satırına yazacağımız o kodun işlevi ile daha açıklayıcı bir hale getiriyoruz. Aşağıdaki örnekte **index.html** dosyasının body elementi içerisine yazacağımız kodları paylaşıyorum.

```
<!--nav element başlangıç-->
<nav>
    <ul>
        <li>Anasayfa</li>
        <li>Kurumsal</li>
        <li>İletişim</li>
    </ul>
</nav>
<!--nav element bitiş-->

<!--section element başlangıç-->
<section>
    <p>Lorem Ipsum, dizgi ve baskı endüstrisinde kullanılan mıgır metinlerdir.</p>
</section>
```

```
<!--section element bitiş-->

<!--footer element başlangıç-->
<footer>
    Copyright © 2021
</footer>
<!--footer element bitiş-->
```

Kaynaklar

- https://www.mehsatek.com/html-yorum-satiri-html-yorum-kodu

HTML Başlıklar ve Paragraflarla Çalışmak

HTML Paragrafları

HTML'de **enter** işlevi yoktur. Bu yüzden eğer farklı farklı paragraflar oluşturmak istiyorsak `<p>` etiketini kullanırız. Bir paragraf her zaman yeni bir satırda başlar ve tarayıcılar bir paragrafın önüne ve arkasına otomatik olarak biraz beyaz boşluk (kenar boşluğu) ekler.

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

HTML Başlıklar

HTML'de en önemli kısım başlıklardır. `<h1>` Etiketi yazıların başlık şeklinde yazılması için kullanılır. Altı tane başlık çeşidi vardır. Bu başlık özellikleri `<h1>` den `<h6>` ya kadar kodlanmıştır. `<h1>` Etiketi en büyük fontu ve en çok önemi bildirirken `<h6>` ya doğru azalma göstermektedir.

```
<h1>H1 Başlık</h1>
<h2>H2 Başlık</h2>
<h3>H3 Başlık</h3>
<h4>H4 Başlık</h4>
<h5>H5 Başlık</h5>
<h6>H6 Başlık</h6>
```

HTML Görünümü

Başlıklar HTML Dökümanlar'da büyük önem arz eder. Arama motorları, web sayfalarınızın yapısını ve içeriğini indekslemek için başlıkları kullanır. Kullanıcılar genellikle başlıkları ile bir sayfayı gözden kaçırırlar. Belge yapısını göstermek için başlıkları kullanmak önemlidir. `<h1>`Başlıklar ana başlıklar için kullanılmalı, ardından başlıklar takip edilmeli `<h3>`, biraz daha az önemli `<h4>`vb.

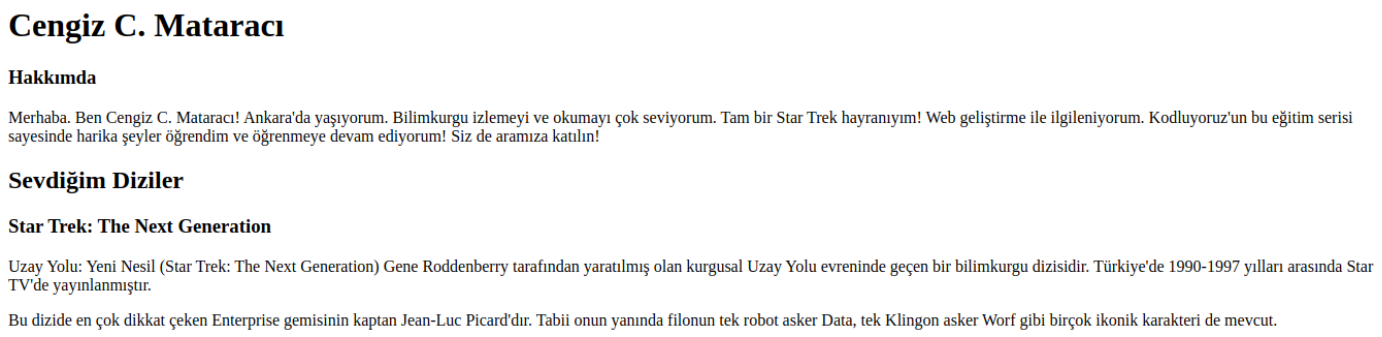
Kaynaklar

- w3schools.com
- html.com

İlk Web Sayfamızı Oluşturmak

Eveet harika konular öğrendikten sonra sıra geldi ödevimize. Bu ödevimizde ilk web sayfamızı tasarlayacağız. Çok heyecanlı değil mi? Sizlerden istediğimiz çok basit bir şekilde öğrendiklerinizle bir web sayfası tasarlamanız.

- Siteyi açtığımızda adınız ve soyadınızı **başlık** şeklinde göstermeniz gerekiyor.
- Ad-Soyadın altında alt başlık olarak **Hakkımda** yazmalıdır.
- Altına paragraf içerisinde neler yaptığınızı ve nelerden hoşlandığınızı yazabilirsiniz.
- Web sitenizi kaydederken dosya adı olarak '**index.html**' seçmeniz gerekmektedir.
- Yazdığınız kodları açıklayan **yorum satırları** eklemeyi unutmayın.



Cengiz C. Mataracı

Hakkımda

Merhaba. Ben Cengiz C. Mataracı! Ankara'da yaşıyorum. Bilimkurgu izlemeyi ve okumayı çok seviyorum. Tam bir Star Trek hayranıyım! Web geliştirme ile ilgileniyorum. Kodluyoruz'un bu eğitim serisi sayesinde harika şeyler öğrendim ve öğrenmeye devam ediyorum! Siz de aramıza katılın!

Sevdiğim Diziler

Star Trek: The Next Generation

Uzay Yolu: Yeni Nesil (Star Trek: The Next Generation) Gene Roddenberry tarafından yaratılmış olan kurgusal Uzay Yolu evreninde geçen bir bilimkurgu dizisidir. Türkiye'de 1990-1997 yılları arasında Star TV'de yayınlanmıştır.

Bu dizide en çok dikkat çeken Enterprise gemisinin kaptan Jean-Luc Picard'dır. Tabii onun yanında filonun tek robot asker Data, tek Klingon asker Worf gibi birçok ikonik karakteri de mevcut.

Yaptığınız sayfa üstteki gibi bir yapıda olabilir.

Bunun haricinde bolca deneme yapmayı, yanlışlar yapıp bunları düzeltmekten çekinmeyin. Unutmayın, doğrular yanlışlar yapılarak bulunur.

Listeleme Etiketleri

Listeleri iki ana başlık altında listeleyebiliriz;

1. Sıralı Listeler
2. Sırasız Listeler

Sıralı Listeler

Sıralı listeler ardışık liste numaraları vermek için kullanılır. Sıralı listelerden yararlanmak için `<ol>` etiketi kullanılır.

```
<p> Gerçek <b>tereyağı</b> nasıl anlaşılır ?</p>
<ol>
    <li>Oda sıcaklığında tamamen erimez</li>
    <li>Suyun içinde tek parça halinde çözünür</li>
    <li>Tereyağı rengi sarı yada beyaz olabilir</li>
</ol>
```

Gerçek **tereyağı** nasıl anlaşılır ?

1. Oda sıcaklığında tamamen erimez
2. Suyun içinde tek parça halinde çözünür
3. Tereyağı rengi sarı yada beyaz olabilir

Ekran çıktısı:

şeklinde olur.

Liste başındaki sıralandırmayı rakamdan başka **roma rakamı veya alfabetik** şeklinde de yapabiliriz. Bunun için **type** özelliğini kullanmamız gerekir.

```
<ol type="I">
  <li>Javascript</li>
  <li>C#</li>
  <li>Php</li>
</ol>
```

Liste başı elmanını roma rakamı ile sıralama

I. Javascript
II. C#
III. Php

Ekran çıktısı:

```
<ol type="A">
  <li>Javascript</li>
  <li>C#</li>
  <li>Php</li>
</ol>
```

Liste başı elemanını alfabetik sıralamak

A. Javascript
B. C#
C. Php

Ekran çıktısı:

`Sırasız Listeler`

Sırasız listeler numaralandırma olmadan oluşturduğumuz listeleredir. Her bir liste elemanı bir satırı kaplayacak şekilde yani blok etiket şeklinde oluşturulur.

```
<ul>
  <li>Çay</li>
  <li>Türk Kahvesi</li>
  <li>Süt</li>
</ul>
```

Ekran çıktısı:

- Çay
- Türk Kahvesi
- Süt

şeklinde olur. Liste elemanlarının başındaki içi dolu daireyi değiştirebilir veya silebiliriz.

**Silmek İçin: **

```
<ul style="list-style-type:none">
  <li>Çay</li>
  <li>Türk Kahvesi</li>
  <li>Süt</li>
</ul>
```

Ekran çıktısı: Çay

Türk Kahvesi

Süt

şeklinde olur. Liste başındaki içi dolu daireyi değiştirmek için ise **disc, square, circle** değerlerini kullanabiliriz.

```
<ul style="list-style-type:square">
  <li>Telefon</li>
  <li>Bilgisayar</li>
  <li>Yazıcı</li>
</ul>
```

Liste başı elemanını kare yapmak

- Telefon
- Bilgisayar
- Yazıcı

Ekran çıktısı:

Kaynaklar **https://www.geeksforgeeks.org/html-li-tag/**

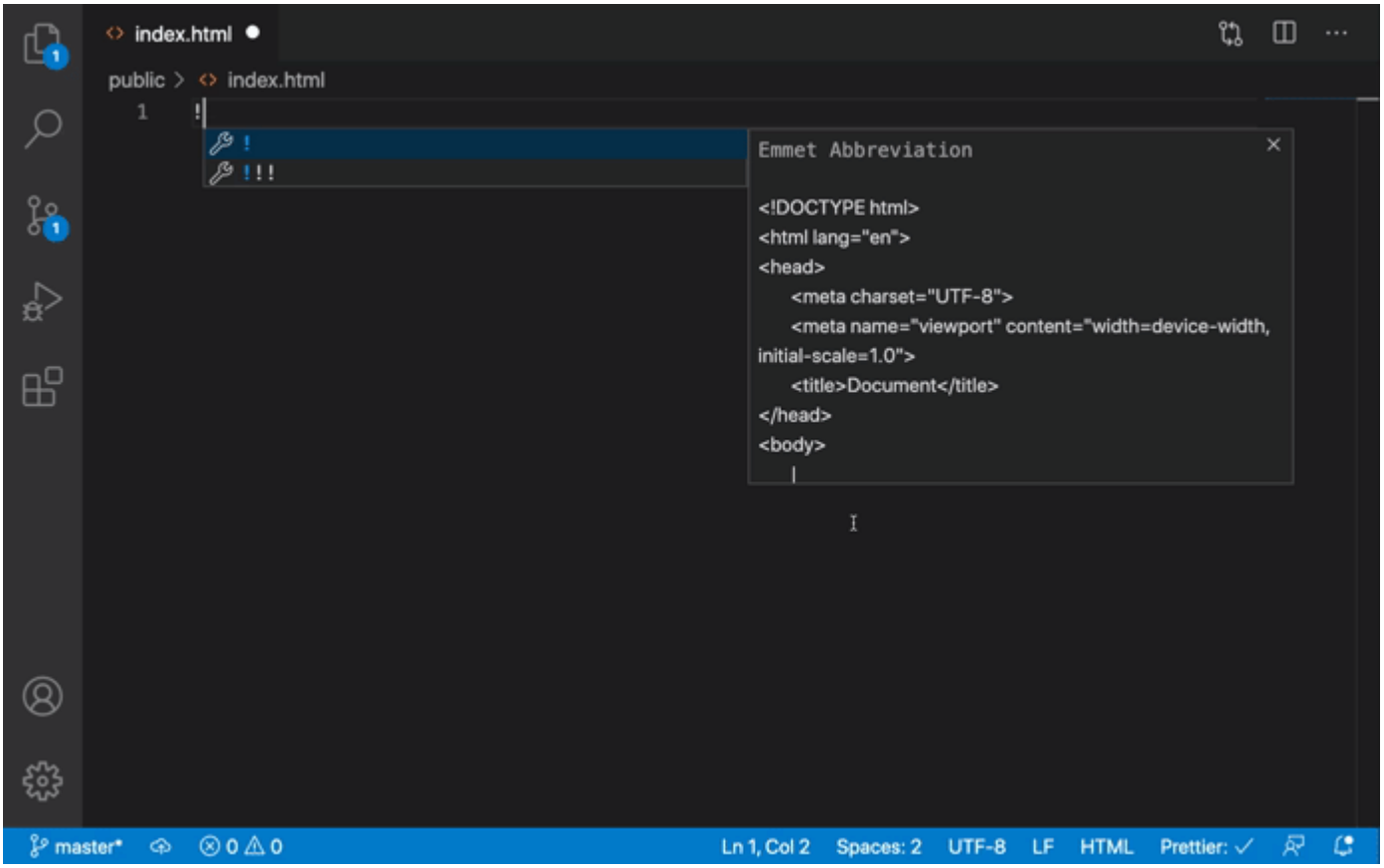
Emmet uzun yıllardır hızlı HTML ve CSS yapıları oluşturmak için kullanılmaktadır. Bu videoda Visual Studio Code içerisinde Emmet kullanımının temellerini öğreneceksiniz. İlerleyen videolarda ise Emmet çok fazla kullanılacağı için Emmet temellerleri konusunda pratik yapmanız oldukça önemli.

Emmet ile Daha Hızlı HTML Yapıları Oluşturmak

Emmet web geliştiricilerinin sıklıkla zamandan tasarruf etmek ve daha hızlı kod yazmak için kullandığı bir eklentidir. Emmet'in temel mantığı, yazılımcıya kodlama yaparken zaman kazandırmasıdır. Örneğin hepimiz bir html dosyasının iskeletini biliriz:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
</body>
</html>
```

Emmet sayesinde çok daha hızlı bir biçimde `!` + Tab kullanarak bu yapıyı oluşturabilirsiniz. Bunu tek tek yazmaktansa iki tuşa basarak yapmak çok güzel değil mi?



Anlayacağınız üzere Emmet bazı kısa yollarla basit bir biçimde HTML ve CSS kodu yazmamıza yardımcı olur. Aynı kodu tekrar tekrar yazmanızı engellerken üretkenliğinizi de arttırmış olur. Emmet neredeyse tüm text editörlerinde mevcuttur, bu yüzden onu yüklemenize gerek yoktur. Ama herhangi bir nedenden IDE'nizde mevcut değilse bu sayfadan yükleyebilirsiniz.

Emmet'deki Kısa Yollara Gelecek Olursak...

Emmette kullandığımız bazı kısa yollar var, şimdi bunları örnekleriyle tek tek inceleyelim.

1 - `>` ifadesini kullanarak kardeş element oluşturuyoruz.

Örneğin şekildeki, gibi `ul` tagı içerisinde `li` tagı oluşturmak istiyorsunuz. Bunun için yapmanız gereken tek şey `ul>li` yazarak Tab'a basmak.

```
<ul>
  <li></li>
</ul>
```

Bu işlemi yaptıktan sonra ul tagına eklemek istediğimiz bir kardeş eleman kalmayınca ise `^` ifadesini kullanarak `ul` tagı dışına çıkıp yeni taglar oluşturabiliriz.

Örneğin `ul` tagı içinde `li` tagı oluşturduktan sonra `ul` tagı dışında bir `p` tagı eklemek istiyorum. Bunun için `ul>li^p` yazarak taba basabilirim.

```
<ul>
    <li></li>
</ul>
<p><p/>
```

2 - Peki ya bu `ul` tagı içerisine birden fazla `li` oluşturmak istiyorsam, ne yapmalıyım?

Bunun için `*` ifadesini kullanırız. `ul>li*3` yaparak `ul` tagı içerisinde üç adet `li` tagı oluşturabilirsiniz.

```
<ul>
  <li></li>
  <li></li>
  <li></li>
</ul>
```

3 - Bunu yerine benzer biçimde kardeş eleman-tag eklemek için `+` ifadesi de kullanılabilirdi: `ul>li+li+li`

4 - Tag eklerken onlara class özelliği vermek için de `.` ifadesini kullanırız.

Örneğin `ul.class1>li.class2` yazılarak tab tuşuna basıldığında:

```
<ul class="class1">
    <li class="class2"></li>
</ul>
```

Bu şekilde bir kod oluşur. Aynı şekilde `ul.class1>li.class2*3` denerek bir yerine üç adet class2 sınıfından `li` tagı oluşturulabilirdi.

5 - Bir id özelliği eklemek için ise `#` ifadesini kullanırız. Yeni bir örnekle id özelliği eklemeyi görelim. `ul#id1>li#id2` diyerek aşağıda gördüğünüz kodu oluşturabiliriz.

```
<ul id="id1">
    <li id="id2"></li>
</ul>
```

6 - Burada `+` ve `*` ifadesinin farkını da daha kolay anlayabiliriz.

Örneğin `ul` tagının içine aynı id'ye sahip 3 adet `li` eklemek istiyorsam `*` ifadesi kullanılabilir.

Fakat amacım farklı `id`'lere sahip üç adet `li` tagı oluşturmaksa `ul>li#id1+li#id2+li#id3` yapılır.

7 - Peki otomatik artış sağlayan değerleri tek tek yazmak amacımız zaman tasarrufu iken ne kadar mantıklı?

Emmet bunun için de bir kısa yola sahip :`$` ifadesi. Yani yukarda görmüş olduğunuz `ul>li#id1+li#id2+li#id3` şeklinde yazdığımız kod bloğunu çok daha basit bir biçimde `ul>li#idNo$*3` diyerek yazabiliriz.

Böylece otomatik olarak `idNo1`, `idNo2` ve `idNo3` idlerine sahip üç adet `li` tagımız olur.

```
<ul>
  <li id="idNo1"></li>
  <li id="idNo2"></li>
```

```
  <li id="idNo3"></li>
</ul>
```

8 - Sırada oluşturulan tagların içine text yazılması var:

Textlerimizi `{}` ifadesinin içine yazmamız yeterli: `p{Emmet ile tag içine text yazma}`

```
<p>Emmet ile tag içine text yazma</p>
```

9 - Bir diğer emmet kısa yolunu ise kodumuzda içerik olmadığı zamanlarda kullandığımız anlamsız yazıları yani **"lorem ipsum"ları** oluşturmak için kullanıyoruz.

Örneğin bir paragraf oluşturacaksınız ve bu paragrafın henüz bi içeriği yok fakat boş durmasındansa oraya metin geleceğini belirtmek istiyorsunuz veya metin geldiğinde nasıl görüneceğini görmek istiyorsunuz. Anlamsız harfler veya zaman gerektiren rastgele cümleler oluşturmak yerine bu kısayolu kullanabilirsiniz : `p>lorem` Taba bastığınızda aşağıdaki gibi bir çıktı alacaksınız.

```
<p>Lorem ipsum dolor sit, amet consectetur adipisicing elit.Facere dolore sint ea? Molestiae ratione ullam, illo commodi ipsum soluta mollitia itaque,maiores maxime natus reiciendis architecto. Quaerat culpa beatae dicta.</p>
```

10 - Bu kadar uzun bir lorem yazısı istemiyorsanız yapmanız gereken tek şey `lorem` yazdıktan sonra yanına kaç kelimeli bir lorem oluşturmak istediğinizi eklemek.

Örneğin 5 kelimeli bir lorem yazısı istiyorsunuz. Bunun için `p>lorem5` yazmanız yeterli.

```
<p>Lorem ipsum dolor sit amet.</p>
```

11 - Son olarak Emmet'in bir özelliğinden bahsedeceğim. `li.className` yazıp Tab'a bastığımızda ne oluşmasını bekleriz? **Evet** `className` class'ına ait bir `li` tagı oluşmasını. peki herhangi bir tag koymaksızın sadece `.className` yazdıktan sonra Tab'a basarsak ne olur?

Cevap:

```
<div class="className"></div>
```

Gördüğünüz gibi bir `div` oluşturdu. Emmet'e bir tag vermeksizin `.` veya `#` ifadelerini kullandığımızda bunun `div` tagı olduğunu biliyor.

Ama biz bunu ul tagı içinde denersek tepkisi ne olur? **Hadi deneyelim!:**

```
<ul>
  <li class="className"></li>
</ul>
```

Gördüğünüz gibi `ul>.className` yazıp Tab'a bastığımızda ise bunun `li` elementi olduğunu algılıyor.

Emmet'in kendi sitesindeki cheat sheete buradan ulaşabilirsiniz. Bu konuda bol bol egzersiz yapmayı unutmayın lütfen, emin olun başta eliniz alışana kadar çok zorlansanız da emmet kullanımı çalışma hızınızı arttıracak ve sizi gereksiz çabadan kurtaracaktır.

Kaynaklar

- Why I love pressing tab, featuring Emmet

Görsellerle Çalışmak

Merhaba arkadaşlar bu yazıda HTML belgesine resim ekleme , bu resimlerle çalışma konusunu inceleyeceğiz. Öncelikle HTML sayfasına resim eklemek için `<img>` tagi kullanılır.

src="" Kullanımı

Kod bloğundaki `src=""` özelliğine görselin URL ya da dosya adresi belirtilerek resim HTML sayfasına çağırılır. (Resmi webden çağırmak için resmin URL'ini src="…" parametesine eklemek yeterlidir.)

```
<img src="ornek.jpg"/>
```

Yukarıdaki örnekte resim HTML dosyasıyla aynı dizinde(klasörde) olduğu için direkt adını ve uzantısını yazmak yeterlidir. Burada img uzantısına dikkat etmek önemli, HTML dosyaları nasıl **.html** ile bitiyorsa tüm resim dosyalarının sonu da **.xbm, .gif, .png veya .jpg** ile veya başka bir resim formatıyla bitmelidir.

Yaygın Image Formatları

Diyelim ki projenin içerisinde bir dizin oluşturdunuz (images) ve resminizi bu dizine eklediniz. Bu defa çağırmak için öncelikle images dizinine gitmek gerekiyor.

```
<img src="images/ornek.jpg"/>
```

Ya da resim bir üst dizinde kalıyor olabilir. Bu durumda bir üst dizine çıkıp images dizinini bulup resmi çağırmak gerekiyor. **(Üst dizine çıkmak için ../ kullanırız.)**

```
<img src="../images/ornek.jpg"/>
```

Bu şekilde istediğiniz kadar üst dizine çıkabilirsiniz.

```
<img src="../../images/ornek.jpg"/>
```

alt="" Kullanımı

Alt textlerin temel amacı, görüntüleri göremeyen kullanıcılar için metinler sunmaktır. Kullanıcı görseli görüntüleyemez ise (Yavaş bağlantı, src özelliğinde hata vb.) alt özelliği görüntü için alternatif bilgilendirici bir metin içerir.

```
<img src="../images/kedi.jpg" alt="Yavru Kedi"/>
```

title="" Kullanımı

Title özelliği kullanıcıyı bilgilendirme amacı taşır. `Cursor`(mouse imleci) ile görselin üzerine gelince bu özelliğe verilen text mesajı görünür. Ek açıklama gerektirecek resimlerde kullanabiliriz. Bilgilendirme amacı taşır.

```
<img src="../images/kedi.jpg" title="image"/>
```

NOT: *Title ve Alt parametreleri SEO açısından önem taşımaktadır.*

width ve height Kullanımı

Görsele istenen ölçüleri vermek için `width` ve `height` özellikleri kullanılır.

Width yatay genişlik, Height ise dikey uzunluk için kullanılır.

Width ve Height özellikleri tanımlanmadığı zaman görsel sayfada gerçek ölçüleri ile görünür. Bu ölçülerden yalnızca birine değer verildiğinde görsel oranları korunarak belirlenen ölçüde görünür. Her iki özelliğe de değer verildiğinde resim oranları yeni verilen ölçülerde olduğu gibi görünür. Bu kullanım resim ölçüleri ile uyumlu olmadığı zaman resimde oransal bozukluklar oluşur.

```
<img src="resim.jpg" width="300" />
<img src="resim.jpg" height="400" />
<img src="resim.jpg" width="300" height="600" />
<img src="resim.jpg" height="100%" />
```

Buradan örneği inceleyebilirsiniz.

`border` Kullanımı

Görseli belirtilen kalınlıkta çerçeve içine alır. Daha gelişmiş **CSS Border** özelliği bunun yerine kullanılabilir.

Aşağıda `border` örneği: Görsele **3 pixel** kalınlıkta border verir.

```
<img src="resim.jpg" border="3" />
```

`align` Kullanımı

Web sayfasında resmin gözükeceği pozisyonu belirlemede align özelliği kullanılır. Bu özelliğe verilebilecek değerler şunlardır: `left`, `right`. Görselin sağa veya sola yaslı çıkmasını sağlar.

```
<img src="resim.jpg" align="right" />
```

Görsele Link Vermek

Görsele link vermek için `img`tag'i `a` taginin içerisinde kullanılır. Yönlendirilmek istenen yerin URL'i `a` taginin `href` özelliğine yazılır. Görsel ise `a` taginin açıklama kısmına eklenir.

```
<a href="default.asp">
  <img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;">
</a>
```

Buradan örneği inceleyebilirsiniz.

`map` ve `area`

Görselleriniz *hyperlink* atamanız durumunda görselin tüm alanı link alanı haline dönüşecektir. Görselin herhangi bir yerine tıklanılması durumunda görsel sizi tanımlanan bağlantıya gönderecektir. `<img>` etiketleri için kullanılan `<map>` ve `<area>` etiketleri ile resmin içindeki koordinatlarla belirlediğimiz bir alanı sadece link haline getirebiliriz. Eklediğimiz `<area>` etiketi kadar belirlenen alanı bir resim üzerinden birçok bağlantıya link verebiliriz.

Örneğin Instagram'da bazı satıcılarda gördüğünüz bir insan fotoğraf üzerinde pantolon ve ayakkabının ürün linklerini ayrı ayrı vermek isterseniz kullanabilirsiniz.

```
<html>
<body>
<img src="workplace.jpg" alt="Workplace" usemap="#workmap" width="400" height="379">
<map name="workmap">
```

```
 <area shape="rect" coords="34,44,270,350" alt="Computer" href="computer.htm">
 <area shape="rect" coords="290,172,333,250" alt="Phone" href="phone.htm">
 <area shape="circle" coords="337,300,44" alt="Cup of coffee" href="coffee.htm">
</map>
</body>
</html>
```

Buradan inceleyebilirsiniz.

`onload` Event'i Kullanımı

Bu olay görsel yüklenmesi tamamlandığında çalışacak fonksiyonu belirler. *Herhangi bir nedenle görsel yüklenemezse ya da belirtilen adreste resim yoksa fonksiyon çalışmaz.*

Aşağıdaki örnekte görsel yüklenmesi tamamlandığında `resimYuklendi()` fonksiyonu çalışacak ve ekrana **Resim Yüklendi.** uyarısı JavaScript tarafından bastırılacak. Bu konuyu ileride çok daha detaylı göreceğiz.

```
<html>
    <body>
        <img src="resim.jpg" onload="resimYuklendi()" />
    </body>
    <script>
        function resimYuklendi(){
            alert("Resim yüklendi.");
        }
    </script>
</html>
```

`picture` Elementi ile Kullanım

HTML5 ile gelen `picture` elementi web sayfamızda responsive image'ler kullanmamız konusunda büyük kolaylıklar sağlıyor. Bir tane `img` ve birden fazla source içerebilir. `picture` tagi ekran boyutlarına göre birden çok source kulllanmamızı sağlar bu sayede ekran boyutu değiştikçe farklı image'leri kullanabilirsiniz. **Örnek:** Burada ekran genişliğinin 800 pikselden küçük olduğu durumlarda başka diğer koşullarda ise başka bir görsel kullanılacak.

```
<picture>
    <source
srcset="https://cdn.sanity.io/images/9kdepi1d/production/65c832d202a503b15d99e628f4313
782f3ef50db-300x62.png" media="(min-width: 800px)">

    <img src="/media/cc0-images/painted-hand-298-332.jpg" alt="" />
</picture>
```

Tarayıcı, her bir source öğesini inceleyip eşleşme sağlar. Eşleşme bulunamazsa veya tarayıcı `<picture>` öğesini desteklemiyorsa, `<img>` öğesinin `src` URL'si seçilir. Seçilen görüntü daha sonra `<img>` öğesinin kapladığı alanda sunulur.

Buradan ekran boyutunuzu değiştirerek inceleyebilirsiniz.

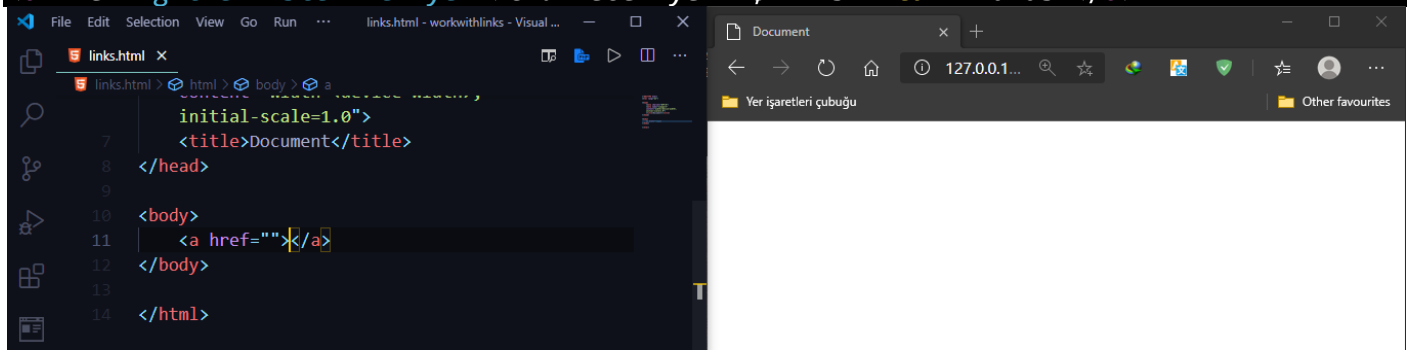
Görsellerle Çalışmak

Yukarıda öğrendiklerinizi uygulayın başka bir görsel olarak

Linklerle Çalışmak

Bir HTML dosyası içerisinde herhangi bir sayfaya, sayfa içine, bir websitesine, e-mail, telefon adreslerine ya da dosya yolu belirtilen herhangi bir dosyaya (resim, video, zip vb.) erişmek için kullanılan HTML etiketidir. Genel yapısı `<a></a>` şeklindedir.

```
<a href="gitmek istenilen yer">Gidilecek yer için isim tanımlaması</a>
```



HTML sayfaları içinde kullanımı:

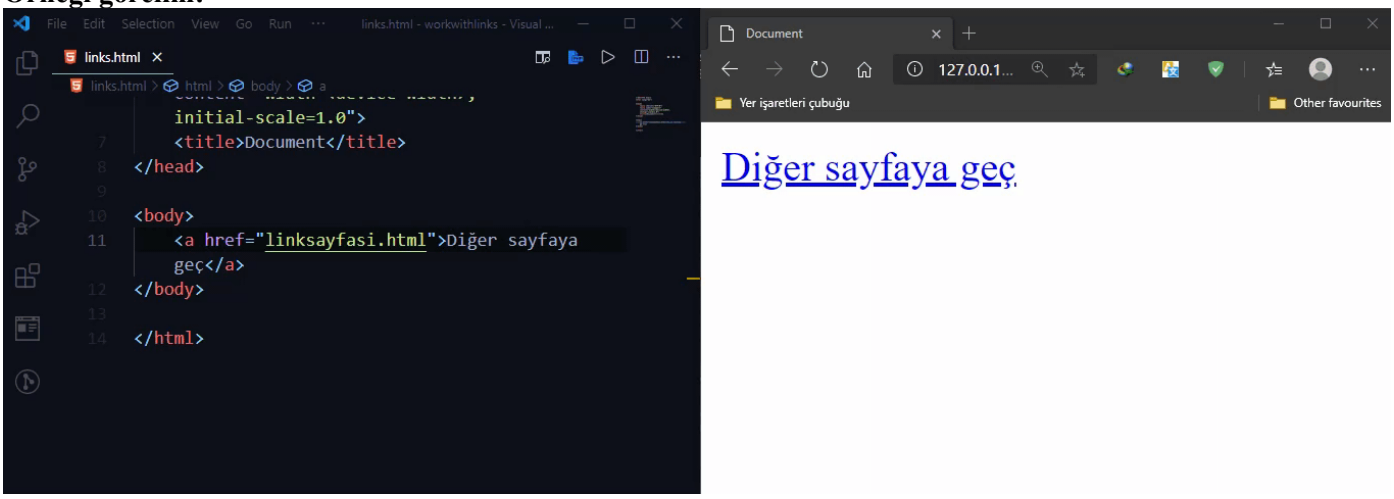
Sayfalar arası kullanımı:

HTML sayfalar arası geçiş yapmak için linklerden şu şekilde faydalanırız. Şu an üzerinde çalıştığımız klasörün içerisinde bir **linksayfasi.html** adında dosya oluşturduk ve içerisine bir şeyler yazdık. Şimdi HTML'de sayfalar arası linkleme ile yeni oluşturduğumuz sayfaya gitmek için link oluşturacağız:

```
<a href="linksayfasi.html">İkinci sayfaya git</a>
```

Bunu oluşturduğumuzda tarayıcı ekranımızda **Diğer sayfaya geç** diye bir link oluşacak (üzerine gelindiğinde mouse'un el işaretine dönmesinden anlaşılabilir)

Örneği görelim:



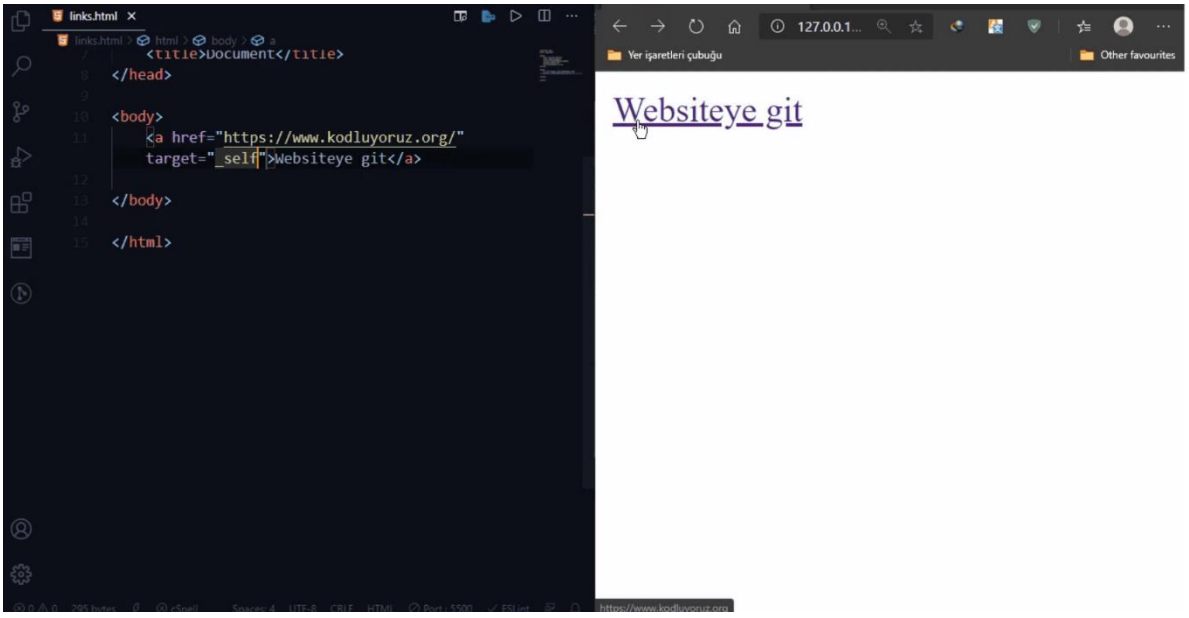
Sayfa içerisinde kullanımı:

Sayfa içerisinde herhangi bir başlığa ya da bölüme gitmek için linkler kullanılabilir. Aynı sayfada işlem yapacağımız için birden fazla satır ekleyeceğim ve en aşağıya scroll ile kaydırmak yerine link yardımı ile gideceğim. Bunun için de gidilecek bölüm için `a` tagine `name` özelliği verilmelidir:

```
<a href="#sonbolum">Aşağı Git</a>
<a name="sonbolum"></a>
```

Website yönlendirmesinde kullanımı:

a etiketinde href özelliğine verilen herhangi bir websitesi adresine kolayca gidilebilir. Burada target özelliğini göreceğiz. Bu özellik, gitmek istediğimiz bağlantının geçerli pencerede mi yoksa yeni bir pencerede mi açılması için kullanılır. _self özelliği geçerli pencerede açılması içindir. Varsayılan olarak böyledir. _blank özelliği ise yeni bir pencerede açılması içindir. **Kodluyoruz**'un internet sitesine gidelim:



Mail ve telefon yönlendirmesinde kullanımı:

a etiketinin href özelliğine verilen mailto: ve tel: özellikleri sayesinde direkt olarak herhangi bir e-mail adresine posta gönderilebilir ya da geçerli bir telefon numarası aranabilir. **Kodluyoruz**'un e-mailine gidelim ve rastgele bir numaraya gidelim.

```
<a href="mailto:info@kodluyoruz.org">Kodluyoruz'a mail atınız.</a>
```

Etiketlerine Ekstra Özellikler Eklemek

Bu ek özellikler Türkçe kaynaklarda HTML etiketlerinin nitelikleri, özellikleri veya öznitelikleri diye geçmektedir.

Özellikleri:

1. Tüm HTML elementleri nitelik alabilirler.
2. Her zaman başlangıç elemanının içine yazılır.

```
<a href = ""></a>
```

1. Bu nitelikler, etiketler konusunda ekstra özellik bilgisi sağlar.
2. Kelimeler arasında tire işareti koyarak(html-etiketi) veya camel case(htmlEtiketi) şeklinde kullanılabilir. HTML'de genellikle - (tire) kullanılır.
3. `class` veya `id` gibi özniteliklerin ismi belirlerken Türkçe karakter tercih edilmemelidir. *(ş,ç,ö,ü,ğ,büyük İ ve küçük ı)*
4. Kullanabilmek için özniteliği yazdıktan sonra `=""` kullanılıp istediğimiz özelliği tırnakların içine yazılmalıdır..

İngilizce bir kaynakta aramak istiyorsanız ***"HTML attributes"*** diye aratmak gerekir.

Videodaki Bilgilerin Özeti:

```
<img src ="" alt = "">
```

HTML dosyasına bir resim eklemek istersek `img` etiketini kullanırız. `img`, İngilizcedeki image(resim) kelimesinin kısaltmasıdır. Bu etiketin yanında ise `src` ve `alt` özniteliklerini ekleyebiliriz. `src` (source = kaynak) ekleyeceğimiz resmi nereden alacağını belirttiğimiz yerdir.

`alt` (alernative = alternatif) ise eğer bir nedenden resim görüntülenemez ise resme alternatif olarak ne yazması gerektiğidir.

```
<a href =""></a>
```

Eğer bir bağlantının URL sini vermek istiyorsak <a> (anchor) etiketini kullanırız. href teki iki tırnağın arası URL'yi girdiğimiz yerdir.

Her etikete `id` ve `class` özniteliklerini ekleyebiliriz.

```
<img src ="" alt = "" id ="">

<img src ="" alt = "" class ="" id ="">

<a href ="" id="" class ="" data-id ="" ></a>
```

id (identity = kimlik)

`id` kullanılan etiketin kimliğini belirtir. **Biriciktir***(unique)* yani dökümanın içindeki bir etikete yalnızca bir kez o `id` ismi verilebilir.

```
<a href ="https://www.facebook.com/" id ="facebook-URL" ><a>
```

Buradaki `id` yi kullanarak direkt olarak bu HTML etiketine ulaşabiliriz. **Not:** `id` biricik olduğu için istisna olarak `id = facebook-URL` şeklinde yazarak tırnak işareti kullanmayabiliriz.

`class` (sınıf)

`class`, HTML etiketinin hangi sınıfta olduğunu belirmemizi sağlar. Aynı sınıf adını **birden fazla** HTML etiketine **verebiliriz.**

```
<img src ="" class ="documents">
<a href ="" class ="documents" ></a>
```

Burada görüldüğü gibi iki HTML etiketi de(`img` ve `a`) documents sınıfında bulunmaktadır. Daha sonrasında aynı sınıfta bulunan bu iki etikete CSS ile aynı anda değişiklik yapılabilir.

Kaynaklar:

- https://www.w3schools.com/html/html_attributes.asp
- https://www.algoritmaornekleri.com/web/html/html-oznitelik-kullanimi-html-attributes/

İskeletinin Genel Yapısını Anlamak

HTML sayfalarının en temel bileşenleri nelerdir? Bir HTML iskeleti nasıl oluşturulur ?

Web tasarım konusunda araştırma yapan hemen herkesin karşına çıkan temel kavram **HTML**'dir. Web sayfalarını oluşturma aşamasında kullanılan standart bir metin işaret dili olan **HTML açılımı "Hyper Text Markup Language"** olarak bilinir. Genel bilinen yanlış kanının aksine **HTML** bir programlama dili değildir.

Daha açık anlatmak gerekirse, Chrome, Firefox, Yandex gibi tarayıcıların okuyup anlamlandırdığı dil **HTML** dilidir.

Bir html sayfası oluşturmadan önce mutlaka **HTML 5 standartlarına uygun bir html iskeleti** oluşturmalıyız. Web sayfalarımızı **HTML 5** standardına uygun hazırlamamız büyük önem taşımaktadır çünkü arama motorları tarafından önemsenmektedir.

HTML iskelet örneği

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Page Title</title>
<meta name="viewport" content="width=device-width,initial-scale=1">
<link rel="stylesheet" href="">
<style>
</style>
<script src=""></script>
</head>
<body>

<img src="img_la.jpg" alt="LA" style="width:100%">

<div class="">
 <h1>Bu bir h1 boyutunda başlık.</h1>
 <p>Bu bir paragraf.</p>
 <p>Bu başka bir paragraf.</p>
</div>

</body>
</html>
```

HTML'inizi Yapılandırma

HTML yapısında sayfanın genel yapısını tanımlamak ve bazı basit başlık bilgilerini sağlamak için 3 farklı etiket bulunur. Bu üç etiket `<html>` , `<head>` ve `<body>` her web sayfasının temel iskeletini oluşturur.

Ayrıca, her şeyi yüklemeden önce sayfa hakkında basit bilgiler *(başlığı veya yazarı gibi)* sağlarlar. Sayfa yapısı etiketleri, sayfanın görüntülendiğinde nasıl görüneceğini etkilemez; sadece tarayıcılara yardımcı olmak için bulunurlar.

DOCTYPE Tanıtıcı

Bir sayfa yapısı etiketi olmasa da, XHTML 1.0 ve HTML 5 standartları web sayfalarınıza ek bir gereklilik getirir. Her sayfanın ilk satırı, sayfanızın uygun olduğu HTML sürümünü tanımlayan bir DOCTYPE tanımlayıcısı ve bazı durumlarda, spesifikasyonu tanımlayan Belge Türü Tanımını (DTD) içermelidir .
Bunu `<html>` , `<head>` ve `<body>` etiketleri takip eder. HTML 5 belge türü, sayfa yapısı etiketlerinden önce görünür.

<html> Etiketi

Her HTML sayfasındaki ilk sayfa yapısı etiketi `<html>` etiketidir. Bu dosyanın içeriğinin HTML dilinde olduğunu gösterir. `<html>` etiketi `DOCTYPE` tanımlamasından hemen sonra kullanılmalıdır. Web sayfanızdaki tüm metin ve HTML öğeleri HTML etiketlerinin başlangıcına ve bitişine yerleştirilmelidir.

`<html>` etiketi sayfasını oluşturan etiketlerin tümünün bir kabı olarak hizmet vermektedir. Bunu dışarıda bırakırsanız, ki bunu yapmamalısınız çünkü bu, sayfanızı geçersiz kılar, tarayıcı sizin için bir `<html>` etiketi oluşturur, böylece sayfa HTML işlemcisi için anlamlı olur.

<head> Etiketi

`<head>` etiketi sayfayla ilgili bilgiler içeren etiketler için bir kapsayıcıdır. Genellikle sayfanın `<head>` bölümünde yalnızca birkaç etiket kullanılır. Sayfanızın hiçbir metnini başlığa (`<head>` etiketleri arasına) koymamalısınız.
Yine `<head>` etiketinin arasında bulunan `lang` anahtar kelimesinin amacı sayfa içeriğinin hangi dilde oluştuğudur. Bunu belirtmemizin sebebi arama motorlarında bize bu sayfayı tarayıcımızın diliyle zıt düştüğü durumda çevirme tavsiyesi verip vermemesini sağlamaktır.

<head> Etiketi Arasında Bulunan Diğer Anahtar Kelimelerin Görevleri

- `<meta>` : Sayfamızı tanımlayan açıklamaların bulunduğu kısımdır. Sayfa açıklaması, sayfa başlığı, kullanılan karakter seti tanımlaması gibi işlemler burada tanımlanıyor.
- `<title>`: Tarayıcı penceresinin sekmesindeki başlığı buradan belirtiyoruz.
- `<style>`: Sayfada kullanılan stil işlemlerinin tanımlaması yapılır. Örneğin arka plan rengi yazı rengi ya da nesnelerin hizalanması gibi işlemler.

<body> Etiketi

HTML sayfanızın içeriği `<body>` etiketi içinde bulunur . Bu, tüm metni ve diğer içeriği *(bağlantılar, resimler vb.)* içerir. Aşağıdaki örnekte etiketlerin iç içe olduğunu göreceksiniz. Yani hem `<body>` ve `</body>` hem de `<head>` ve `</head>` etiketlerini `<html>` etiketlerinin kapsadığını göreceksiniz. Tüm HTML etiketleri bu şekilde çalışır ve metnin tek tek iç içe bölümlerini oluşturur.

Örnek:

```
<! DOCTYPE html>
<html>
<head>
<title> Bu bir başlıktır.</title>
</head>
<body>
... sayfanız ...
</body>
</html>
```

Etiketleri asla çakıştırmamaya dikkat etmelisiniz. Yani, asla aşağıdaki gibi bir şey yapmayın:

```
<! DOCTYPE html>
<html>
<head>
<body>
</head>
</body>
</html>
```

Bir HTML etiketini her kapatışınızda, en son kapatılmamış etiketi kapattığınızdan emin olun.

Kaynaklar

- Learning the Basics of HTML informit.com
- Using an HTML Skeleton w3schools.com

Semantic(Anlamlandırılmış) HTML Etiketleri Kullanımı

Semantik, anlam veya anlamlandırma anlamı ifadesi taşımaktadır. O halde semantik elementler herhangi bir anlamı olan etiketler ifadesi taşımaktadır. Semantik olarak anlamlandırılmış bir element hem tarayıcıya hem geliştiriciye ne anlama geldiğini açık bir şekilde belirtir. `<div>` ve `<span>` gibi elementler semantik olmayan elementlerdir ve mevcut içeriğin hakkında bilgi vermezler. `<form>`, `<table>` ve `<img>` gibi elementler semantik elementlerdir ve içeriği açıkça belirtirler.

header Elementi

`header` elementi bir doküman veya bir `<section>` için bir başlık olduğunu belirtir. İçinde barındırdığı içeriği **kapsayıcı** olmalıdır. Bir dokümanda birden fazla kullanılabilir.

Aşağıdaki örnek, bir `<article>` için bir başlık içermektedir.

```
<header>
    <h1>Kodluyoruz Akademi Nedir?</h1>
</header>
<p>Her bireyin, özellikle kadınların, yükselen teknoloji sektöründe başarılı olması
için eşit haklara sahip olması gerektiğine inanıyoruz.
Bu yüzden, Kodluyoruz Akademi gençlere dünya çapında kaliteli ve ücretsiz içerik,
kaynak ve bootcamp sağlıyor!</p>
```

nav Elementi

`nav` elementi navigasyon bağlantıları büyük sayfalar için ortaya çıkarılmıştır. Fakat, sayfadaki tüm linkler bu element içinde olmak zorunda **değildir**.

```
<nav> Bootcamp CS50X Kodluyoruz Jr. Şirketler Hakkımızda </nav>
```

<section> Elementi

Section elementi bir doküman içinde olan sadece bir kısmı belirtir.

```
<section>
<h1>CS50xKodluyoruz Challenge Başlıyor!</h1>
<p>CS50xKodluyoruz ödevlerine devam etmekte zorlanıyor musun? Bitirmek istediğin halde
nasıl ilerleyeceğini bilmiyor musun?
O zaman CS50xKodluyoruz Challenge tam sana göre! Challenge ekibiyle hedefimiz: 4.
haftaya kadar bütün ödevleri tamamlamak olacak!</p>
</section>
<section>
<h1>CS50x Nedir?</h1>
<p>CS50x (Computer Science 50), Harvard Üniversitesi Profesörü David J. Malan
tarafından verilen efsanevi bilgisayar bilimlerine giriş kursu.
Her yıl milyonlarca kişi tarafından alınan bu kursu Türkçe'ye çevirsek ve Türkiye'nin
her yerinden gençlere ulaştırsak nasıl olur?
Üstelik online ve ücretsiz olarak? Üstelik çalışma gruplarında her yaştan gencin
beraber öğrenmesini destekleyerek?</p>
</section>
```

figure Elementi

İçeriğinde resim, gösterim, diyagram, kod listeleri vs. gibi nesnelerin olduğunu belirtir. Ana akış ile ilgili olsa da, konumu ana akıştan tamamen bağımsızdır. Çıkarılırsa dokümanın akışını **engellemez**.

```
<p>CS50x Kodluyoruz için hemen kayıt ol, dersleri anında almaya başla. Dersler tamamen online ve ücretsiz! Üstelik CS50x Kodluyoruz herkese göre.
İster hiç bilgisayar dersi almamış olun, ister kendinizi ilerletmek isteyin: Bu ders, sağlam bir algoritma temeli isteyen herkes için!</p>

<figure>
        <img
src="https://cdn.sanity.io/images/9kdepi1d/production/1a22b9fada2411b7369592fd852613637834e866-5520x3680.jpg?auto=format" />
</figure>
```

figcaption Elementi

<figcaption> etiketi, <figure> elementinin belirttiği resme başlık koymaya yarar.

```
<figcaption>
        <p>A duck.</p>
        <p><small>Photograph courtesy of News.</small></p>
</figcaption>
```

aside Elementi

<aside> elementi içerdiğinden farklı olarak daha başka bazı içerikleri tanımlar. İçeriği, üst içerik hakkında olmalıdır.

article Elementi

<article> elementi bir makale elementidir. Bir makale web sayfasının geri kalanından bağımsız olarak dağıtılabilmelidir.

Genelde bu elementin kullanabildiği yerler forum mesajları, blog gönderileri, haber metinleri, yorumlar gibi makale içeren metinlerdir.

footer Elementi

<footer> elementi bir doküman ya da kısım için alt bilgilerini belirtir. Bir<footer> genelde dokümanın yazarını, telif haklarını, kullanım gizliği, iletisim vs. gibi bilgileri içerir ve bir dokümanda bir kereden fazla kullanılabilir.

Diğer HTML Etiketleri Hakkında

HTML öğrenirken sıkça sorulan bir soru tüm etiketlerin öğrenilmesi gerekip gerekmediğidir. HTML güncel olarak 100'den fazla etiket içerir ancak bunların hepsini öğrenmek mümkün değildir. Günümüzde birçok HTML sayfası semantik elementler ve 10-12 adet semantik olmayan etiket ile oluşturulabilir.

h1 - Başlık

p - Paragraf

i / em - Eğik / Vurgulu Yazı

b / strong - Kalın / Koyu Yazı

a - Link

ul & li - Sırasız liste ve liste elemanı

blockquote - Alıntı

hr - Yatay çizgi ekleme

img - Görsel

div - Bölme

Yukarıda yaygın kullanılan semantik olmayan etiketler ve görevleri verilmiştir. Bunlar dışında bir etikete ihtiyaç duyulduğunda istenen fonksiyona dair yapılacak araştırmayla ilgili etikete ulaşılabilir.

Emmet

HTML ve CSS etiketlerinde kullanım kolaylığı sağlaması açısından **Emmet** oldukça faydalı bir eklentidir. Kısaltılmış kod yapıları yardımıyla uzun kod blokları oluşturmayı sağlar.

Asıl koda dönüşmeden önceki Emmet yapısı:

```
<!doctype html>
<html lang ="en">
<head>
  <title>Demo</title>
</head>
<body>
  ul#nav>li.item$*4>a{Item $}
</body>
</html>
```

Emmet sonrası kodun tam formu:

```
<!doctype html>
<html lang ="en">
<head>
  <title>Demo</title>
</head>
<body>
  <ul id="nav>
    <li class="item1"><a href="">Item 1</a></li>
    <li class="item2"><a href="">Item 2</a></li>
    <li class="item3"><a href="">Item 3</a></li>
    <li class="item4"><a href="">Item 4</a></li>
       </ul>
</body>
</html>
```

Kaynaklar

- http://www.99lime.com/_bak/topics/you-only-need-10-tags/
- https://emmet.io/
- https://www.themt.co/blog/5-bilisim/353-emmet-nedir-ve-ne-ise-yarar