CSS Dersi Notlarım

CSS Nedir?

CSS

CSS'de arka plan renklerini değiştirmekten ve nesnelerin neden olması gerektiği gibi hizalanmadığını merak etmekten daha fazlası vardır. Ancak JavaScript framework'leri ve kütüphaneleri gibi diğer araçlara zaman ayırdığımız için, CSS genellikle gözden kaçan ve arka planda kalan görsel bir iletişim aracıdır. Genellikle temel çalışma yeterliliğine sahip CSS'i alırız ve acil hedeflerimize ulaşır ulaşmaz daha derine inmeden bırakırız. Sonuç olarak uygulama genelinde yapı, mimari ve uyum gibi şeyleri göz ardı etme eğilimindeyiz. Biz bu yazımızda CSS hakkında bilmemiz gerekenleri ele alacağız ve şimdi en baştan başlayalım.

CSS(Cascading Style Sheets) Nedir?

CSS (Basamaklanmış Stil Katmanları), web sayfasını şekillendirmek için kullanılan bir kodlama türüdür. CSS kullanarak web sayfasının görünümünü ve düzenini değiştirebiliriz. Bir web sitesinin görünümünün masaüstü bilgisayarlar, tabletler ve mobil cihazlar gibi farklı ekranlarda nasıl değiştiğini de tanımlayabiliriz. CSS, *C++ veya JavaScript* gibi bir programlama dili **değildir**. Bunun nedeni, CSS'in amacının web tarayıcıları için biçimlendirmeye (HTML'e) stil vermek olmasıdır. HTML yalnızca içeriği "işaretleyen" bir dildir - yani, belgeye belirli bir görsel ve yapısal biçimlendirme verir.

Hem HTML hem de CSS doğaları gereği bildirimseldir. Bu, yalnızca bir şeyleri açıkladığı ve herhangi bir işlem yapmadığı anlamına gelir. Tarayıcılar CSS'i görüntüleme için yorumlarken, bunu doğrusal bir şekilde yapar. Bunun nedeni, CSS'in dosyayı yukarıdan aşağıya okuyan tarayıcıya talimatlar vermesidir. Bir kural belirlendiğinde ve daha sonra dizin aşağıya doğru değiştirildiğinde, tarayıcı ekranda görüntülenecek nihai sonuç olarak en son değerlendirmeyi alır. CSS'deki C'nin ifade ettiği şey budur. CSS'in cascade(kaskad)etkisi, tarayıcıyı, kuralın fiziksel olarak ne zaman gerçekleştiğine bağlı olarak, kuralları art arda ve geçersiz kılacak şekilde görüntüleyecek şekilde ayarlar. Cascade(kaskad), öğe için birden çok CSS bildirimi ve birden çok stil sayfasının bir kombinasyonu olduğunda, belirli bir öğe için uygulanan stiller arasındaki çakışmayı çözme anlamına gelir. Sonuç olarak, CSS göründüğü kadar kolay değildir. Anlamadan kullanmaya çalışırsanız web geliştirmede zorluk yaşarsınız. Bu nedenle, CSS öğrenmek bir programlama dili öğrenmek kadar önemlidir.

CSS Syntax

İlk CSS kodumuzu yazalım. Örneğin, h1 etiketinin rengini değiştirmek istiyorum.

<h1>I'm a Header</h1>



Öncelikle, CSS'e HTML öğesini nasıl bulacağını söylememiz gerekir. Bunu selector denen bir özellik ile yapabiliriz. CSS'de bir selector, HTML öğelerini etiket adı, sınıf adı, kimliği ve çok daha fazlasına göre bulmak için kullanılır.

O zaman şimdi HTML öğesi için bir selector tanımlayalım ve ardından selector parantezleri arasında her biri noktalı virgülle biten CSS kurallarını bildirelim.

h1 { color: red; }



Böylelikle, tanımlı selector'e (h1) bağlı olarak, CSS artık yeni kuralların nereye uygulanacağını anlayabilir.

Bunların yanı sıra CSS kodlarımızı .css uzantılı bir dosya içerisinde tutar ve bu dosyayı HTML sayfası içerisine basit bir kodla çağırabiliriz. Örneğin CSS dosyamızın adı **"hello.css"** olsun ve bu dosyayı Html sayfaya eklemek için HTML içine yazmamız gereken kod şu şekildedir.

<head> <link rel=”stylesheet” type=”text/css” href=”hello.css” /> </head>

Bir CSS selector tanımlamanın birçok farklı yolu vardır, aşağıda selector türlerinin bazı örneklerini görebilirsiniz:

* **Class Selector:** HTML öğelerini sınıf özelliklerine göre bulur.
* **ID Selector:** Öğeleri belirli kimliğine göre bulur.
* **Element Selector:** Öğeleri etiket adlarına göre bulur.

CSS teknik olarak kolaydır. Anlaşılması gereken sınırlı sayıda kural vardır. Bununla birlikte, CSS'i zorlaştıran potansiyel kural kombinasyonudur. Görsel gereksinimler daha karmaşık hale geldikçe, CSS kurallarınızın sürdürülebilirliğini kontrol altında tutmak için yaratıcı ancak yapılandırılmış çözümlere ihtiyaç vardır.

Tasarım yönleri pazarlamaya ve UX tasarımcısına bırakılabilirken, front-end geliştiricilerin tasarımın nasıl çalıştığını anlamaları da iyidir. Tasarımcıların tümü geliştirici değildir ve bazen tarayıcıların ve piksel rehberliğinin daha ince gereksinimlerini anlamayanlarla karşılaşıyoruz. Dijital tasarımcıların stil sayfalarının inceliklerini ve potansiyel karmaşıklığını öğrenmesi gerektiği kadar, tasarım dilinde de konuşabilmelisiniz.

**Yukarıda anlatılanları aşağıda Codepen içerisinde deneyimleyebilirsiniz!**

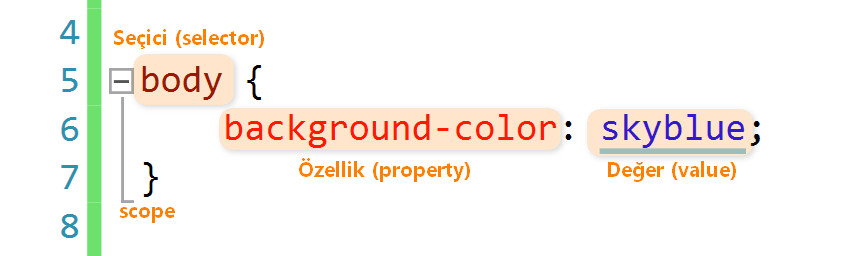
Kaynaklar

* <https://levelup.gitconnected.com/what-is-css-specificity-and-how-it-works-833bb0b3d3a>
* <https://blog.isimtescil.net/css-nedir/>
* <https://medium.com/swlh/css-for-beginners-what-is-css-and-how-to-use-it-in-web-development-5985afe53096>
* <https://medium.com/madhash/what-exactly-is-css-d21d17f1031f>
* <https://www.vargonen.com/blog/css-nedir-css-kodlari-nelerdir/>
* <https://weebakademi.com/html-sayfaya-css-ekleme.aspx>

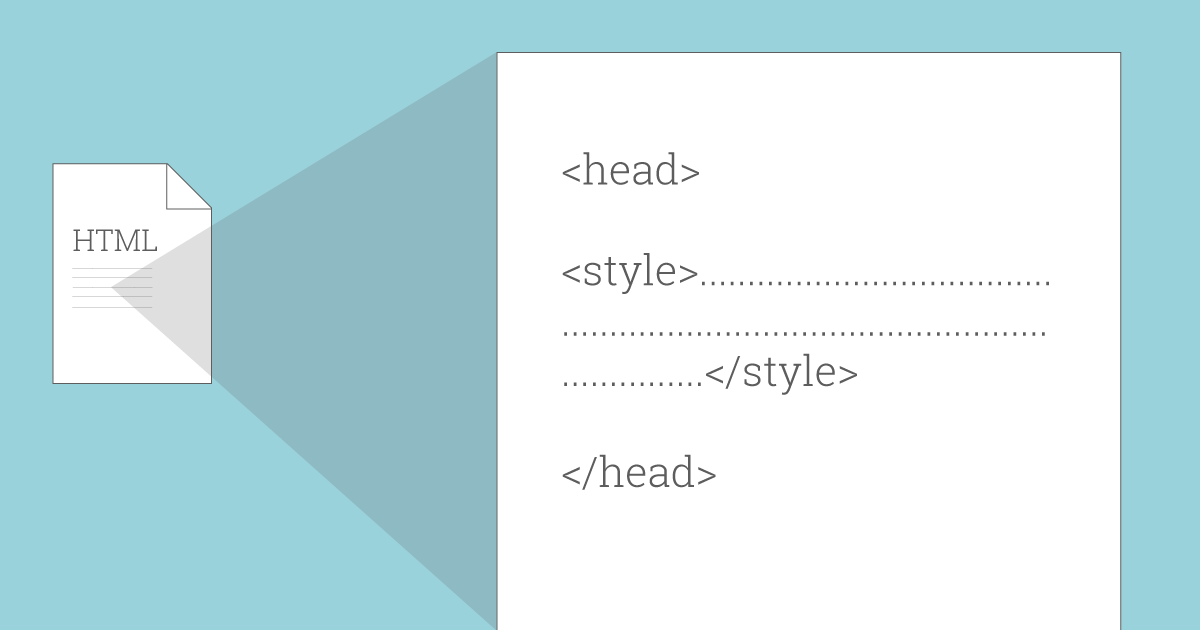
# External, Internal ve Inline CSS Kullanımı

CSS, web sitenizin ekranda nasıl görüneceğini belirleyen belirli stil kurallarına sahip dosyalardır. CSS kuralları, web sitenizin HTML dosyalarına çeşitli şekillerde ve yerlerde uygulanabilir. External yani harici bir stil sayfası, Internal yani dahili bir stil sayfası veya Inline yani satır içi stil kullanabilirsiniz. Her yöntemin belirli kullanımlara uygun avantajları vardır.

External stil sayfası, bir web sayfasından bağlanan bağımsız bir .css dosyasıdır. External stil sayfasının avantajı, bir kez oluşturulabilmesi ve birden çok web sayfasına uygulanabilmesidir. Site tasarımınızda geniş çaplı değişiklikler yapmanız gerekirse eğer, stil sayfasında tek bir değişiklik yapabilirsiniz ve bu değişiklik tüm bağlantılı sayfalara uygulanarak zamandan ve emekten tasarruf sağlar.



Internal stil sayfası, HTML dosyasının <head> bölümünde sayfa için CSS kural kodlarını barındırır. Kural kodları yalnızca o sayfa için geçerlidir, ancak sayfa kodunda birden çok öğeye stil uygulamak için kullanılabilecek class ve id’leri yapılandırabilirsiniz. Yine, CSS kodunda yapılacak tek bir değişiklik, sayfadaki tüm etiketlenmiş öğelere uygulanacaktır.



Inline stili, belirli bir sayfa öğesine stil vermek için, belirli bir HTML etiketi içinde kullanılır. Hızlı ve kalıcı değişiklikler için kullanışlıdırlar, ancak bir tasarım değişikliği yapmaya karar vermeniz durumunda, oluşturduğunuz her Inline stilin ayrı ayrı düzenlenmesi gerektiğinden, External ve Internal stil sayfalarından daha kullanışlı değildir ve diğerlerine göre çok fazla tercih edilmez.

## External CSS Kullanımı

Yukarıda bahsettiğimiz gibi External CSS, normal HTML dosyanızdan bağımsız bir dosyadır ve .css uzantılıdır. CSS dosyanızı oluşturduktan sonra mutlaka HTML dosyanıza linklemeniz gerekir. Yoksa CSS dosyanız işlevsiz kalacak ve HTML sayfanızda yaptığınız değişiklikleri göremeyeceksiniz. Hadi gelin bir de örnek üzerinde görelim.

.css uzantılı dosyamızı oluşturduktan sonra içine kodlarımızı bu şekilde yazıyoruz:

body {

background-color: #f7cac9;

}

.firstDiv, .secondDiv, .thirdDiv {

border: 7px outset gold;

background-color: lightblue;

text-align: center;

}

Daha sonra HTML sayfamıza geri dönüyoruz ve yazdığımız bu External CSS kodunu <head> bölümüne aşağıdaki gibi linkliyoruz. İşte bu kadar. :smile:

<link rel="stylesheet" href="style.css">

Aşağıda codepen ile deneyimleyebilirsiniz!

## Internal CSS Kullanımı

Internal CSS, stil kodlarının direk HTML dosyasının içinde <head> bölümünde kullanıldığı bir yöntemdir. <head> bölümünün içinde bir <style> etiketi açtıktan sonra içine gerekli CSS kodlarını yazabilirsiniz.

**Örnek vermek gerekirse; kodunuz şu şekilde görünecektir:**

<!DOCTYPE html>

<html>

<head>

<style>

body {

background-color: #f7cac9;

}

.firstDiv {

border: 7px outset gold;

background-color: lightblue;

text-align: center;

}

</style>

</head>

<body>

<div class="firstDiv">

<h2>Lorem ipsum</h2>

<p>Lorem ipsum dolor sit amet consectetuer</p>

</div>

</body>

</html>

**Aşağıda codepen ile deneyimleyebilirsiniz!**

## Inline CSS Kullanımı

Inline yani satır içi stiller, doğrudan HTML kodunuzdaki herhangi bir öğeye uygulanır. Stil niteliği ve ardından normal CSS özellikleri bir HTML etiketi içinde belirtilir. Kullanımı aşağıdaki örnekte gösterildiği gibidir:

<!DOCTYPE html>

<html>

<head>

</head>

<body>

<div style="border-width:7px; border-style:outset; border-color:gold; background-color:lightblue; text-align:center;">

<h2>Lorem ipsum</h2>

<p>Lorem ipsum dolor sit amet consectetuer</p>

</div>

</body>

</html>

**Aşağıda codepen ile deneyimleyebilirsiniz!**

Bu üç CSS özelliğinin nasıl kullanıldığını öğrendiğinize göre aşağıdaki alıştırma ve iki adet çoktan seçmeli soruyla öğrendiğiniz bilgileri pekiştirebilirsiniz. :smile:

## Ekstra Alıştırmalar

Yapabileceğiniz ekstra alıştırmalar.

* [Alıştırma 1 için tıkla](https://www.w3schools.com/css/exercise.asp?filename=exercise_howto2)
* [Alıştırma 2 için tıkla](https://www.w3schools.com/css/exercise.asp?filename=exercise_howto3)
* [Alıştırma 3 için tıkla](https://www.w3schools.com/css/exercise.asp?filename=exercise_howto4)

## Kaynaklar

* <https://www.w3schools.com/css/exercise.asp?filename=exercise_howto2>
* <https://www.w3schools.com/css/exercise.asp?filename=exercise_howto3>
* <https://www.w3schools.com/css/exercise.asp?filename=exercise_howto4>
* <http://zinzinzibidi.com/web_tasarim/css3/css_in_yapisi>
* <https://varvy.com/inline-small-css.html>

# Açıklama Satırları ile Çalışmak ve Genel Font Özellikleri

## Açıklama Satırları

Açıklama satırları CSS kodu yazarken aralara kendimiz ve diğer yazılımcılar için notlar bırakmamızı sağlar. Açıklama satırlarının önemi kodda güncelleme yapılacığı vakit anlaşılmaktadır. Örnek olarak 5000 satır css kodunuz olduğu düşünün, kodu yazdıktan 1 yıl sonra değişiklik yapmak istediğimizde nerede ne yazdığınızı hatırlamamız çok zordur. Bunun için açıklama satırlarını okuruz ve bulmak istediğimiz yerleri daha kolay buluruz. Açıklama satırları tarayıcılar tarafından görünmemektedir.

CSS'de açıklama satırı bu şekilde başlar /\* ve bu şekilde biter \*/

**Örneğin**

/\* Tekli yorum satırı \*/

p {

color: red;

}

Kod da istediğiniz yere yorum satırı ekleyebilirsiniz:

p {

color: red; /\* Metin rengini kırmızı yap\*/

}

Yorumları birden çok satıra da yazabilirsiniz:

/\* Çoklu yorum

satırı bu

şekilde \*/

p {

color: red;

}

## Genel Font Özellikleri

CSS'de beş genel yazı tipi ailesi vardır:

1. **Serif** yazı tiplerinin her harfin kenarlarında küçük bir kontur vardır. Bir formalite ve zarafet duygusu yaratırlar.
2. **Sans-serif** yazı tiplerinin temiz satırları vardır (küçük konturlar eklenmez). Modern ve minimalist bir görünüm yaratırlar.
3. **Monospace** yazı tipleri burada tüm harfler aynı sabit genişliğe sahiptir. Mekanik bir görünüm yaratırlar.
4. **Cursive** yazı insan el yazısı taklidi gibidir.
5. **Fantasy** yazı tipleri dekoratif, eğlenceli yazı tipleridir.

Tüm farklı yazı tipi adları, genel yazı tipi ailelerinden birine aittir.

**Not:** Bilgisayar ekranlarında, sans-serif yazı tiplerinin serif yazı tiplerinden daha kolay okunabileceği kabul edilir.

### **CSS'de, font-family bir metnin yazı tipini belirtmek için özelliği kullanırız.**

**Not:** Yazı tipi adı birden fazla kelime ise, "Times New Roman" gibi tırnak içinde olmalıdır.

.p1 {

font-family: "Times New Roman";

}

.p2 {

font-family: Arial;

}

.p3 {

font-family: monospace;

}

#### **font-style çoğunlukla italik metnini belirtmek için kullanılır.**

* **normal** Metin normal şekilde gösterilir.
* **italic** Metin italik olarak gösterilir.
* **oblique** Metin eğimli gösterilir ama çok desteklenmiyor.

p.normal {

font-style: normal;

}

p.italic {

font-style: italic;

}

p.oblique {

font-style: oblique;

}

#### **CSS'de font-size metnin boyutunu ayarlar.**

Metin boyutunu yönetebilmek web tasarımında önemlidir. Bununla birlikte, paragrafların başlık gibi veya başlıkların paragraflara benzemesi için yazı tipi boyutunu ayarlamalısınız. Bunun içinde font-size kullanılır.

**Not:** Bir yazı tipi boyutu belirtmezseniz, paragraflar gibi normal metin için varsayılan boyut 16 pikseldir (16px = 1em).

h1 {

font-size: 40px;

}

h2 {

font-size: 30px;

}

p {

font-size: 14px;

}

#### **Em ile Yazı Tipi Boyutunu Ayarla**

Kullanıcıların metni yeniden boyutlandırmasına izin vermek için (tarayıcı menüsünde), birçok geliştirici piksel yerine em kullanır.

1em, mevcut yazı tipi boyutuna eşittir. Tarayıcılarda varsayılan metin boyutu 16 pikseldir. Yani, 1em'in varsayılan boyutu 16 pikseldir.

h1 {

font-size: 2.5em; /\* 40px/16=2.5em \*/

}

h2 {

font-size: 1.875em; /\* 30px/16=1.875em \*/

}

p {

font-size: 0.875em; /\* 14px/16=0.875em \*/

}

Yukarıdaki örnekte, em içindeki metin boyutu, piksel cinsinden önceki örnekle aynıdır. Ancak em boyutu ile tüm tarayıcılarda metin boyutunu ayarlamak mümkündür.

## Kaynaklar

* [Yorum satırı kaynak](https://www.w3schools.com/css/css_comments.asp)
* [Font-family kaynak](https://www.w3schools.com/css/css_font.asp)

# Class ve ID Kullanımı

Web sayfamızı oluştururken HTML elementlerimize bazı stil özellikleri eklemek isteriz. Bazı yazıların renkli, bazı resimlerin küçük veya bazı butonların farklı şekilde olması gerekebilir ve biz de bunun için CSS kullanırız.

Aşağıda html elementlerine nasıl stil özellikleri eklediğimize bakalım.

<p>Bugün kodluyoruz</p>

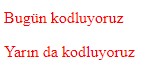
<p>Yarın da kodluyoruz<p>

p {

color: red;

}

**Sonuç**:



Yukarıda görüldüğü üzere iki farklı <p></p> elementimize kırmızı renk özelliği eklemiş olduk. Fakat sadece belirli <p></p> elementine istediğimiz herhangi bir özelliği eklemek istersek ne yapacağız? Bu durumda class veya id seçicilerini kullanmamız gerekiyor.

# class Kullanımı

Class seçicisi, HTML üzerinde aynı class’a sahip elemana ulaşmamızı sağlar.

Class seçicisi CSS’de . ile belirtilir.

.class{

özellikler

}

<h4>Birinci Başlık</h4>

<h4>İkinci Başlık</h4>

<h4>Üçüncü başlık</h4>

Yukarıda 3 adet <h4></h4> elementimiz bulunuyor. Bunlardan sadece ikincisine kırmızı renk özelliği eklemek istersek class seçicisini kullanabiliriz.

<h4>Birinci Başlık</h4>

<h4 class="text-red">İkinci Başlık</h4>

<h4>Üçüncü Başlık</h4>

Daha sonra bunu istediğimiz özelliği ekleyelim.

.text-red{

color:red;

}

**Sonuç:**



Bir class’ı birden fazla HTML elementi için kullanabiliriz.

<h4>Birinci Başlık</h4>

<h4 class="h-green">İkinci Başlık</h4>

<h4 class="h-green">Üçüncü Başlık</h4>

.h-green {

color:green;

}

**Sonuç:**



Eğer bir HTML elementinin birden fazla class özelliğine sahip olmasını istiyorsak aynı anda iki farklı class’ı kullanabiliriz. Bunun için sadece iki class arasına boşluk bırakmak yeterli olacaktır.

<h4>Birinci Başlık</h4>

<h4 class="h-blue">İkinci Başlık</h4>

<h4 class="h-blue thick">Üçüncü Başlık</h4>

.h-blue{

color:blue;

}

.thick{

font-style: italic;

}

**Sonuç:**



Bir HTML elementi kendini kapsayan elementin (parent elementi) stil özelliklerine sahip olur.

<div class="intro"><p>Birinci Paragraf</p><p>İkinci Paragraf</p>

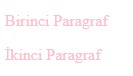
</div>

.intro{

color:pink;

}

**Sonuç:**



Yukarıda <div></div> elementine CSS özelliği ekledik fakat altındaki elementler (child elementleri) de bu özelliğe sahip oldu.

# id Kullanımı

ID seçicisi, HTML üzerinde aynı id’ye sahip elemana ulaşmamızı sağlar.

ID seçicisi CSS’de # ile belirtilir.

ID seçicisinin kullanım amacı olarak class seçicisinden bir farkı yok diyebiliriz. İkisi de belirli HTML elementlerine CSS özellikleri eklemeye yöneliktir. Fakat id seçicisinin class seçicisinden bazı farkları vardır.

#id {

özellikler

}

Bir id’yi sadece bir HTML elementi üzerinde kullanabiliriz.

<h4 id="main-title">Birinci Başlık</h4>

<h4>İkinci Başlık</h4>

#main-title{

color:red;

}

**Sonuç:**

Aşağıdaki **yanlış** bir kullanımdır!

<p id="title">Birinci paragraf</p>

<p id="title">İkinci paragraf</p>

Bir html elementinin sadece bir tane id’si olabilir.

Aşağıdakiler **yanlış** kullanımlardır.

<p id="title" id="text-green">Birinci paragraf</p>

<p id="title text-blue">İkinci paragraf</p>

Böyle kullanımlar geçerli **değildir.**

## Kaynaklar

* [W3Schools](https://www.w3schools.com/)

# CSS Seçiciler ile Çalışmak, İstediğimiz HTML Etiket Yapısına Özellik Ekleyebilmek

CSS seçiciler, HTML taglarına ulaşarak biçim atamaları yapmamızı sağlar. CSS işlemleri, HTML tagı içinde style attribute kullanarak yani inline, <head> arasında <style> tagı kullanarak yani internal ya da dosya uzantısı .css olan bir dosya oluşturup bu dosyayı <head> arasına ekleyerek yani external şekilde kullanabiliriz.

## Inline

<h1 style="color:orange;">Turuncu renkli başlık</h1>

## Internal

<head>

<style>

h1{

color: orange;

}

</style>

</head>

<body>

<h1>Turuncu renkli başlık</h1>

</body>

## External

<head>

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

</head>

<body>

<h1>This is a heading</h1>

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

</body>

styles.css dosyası:

h1{

color:orange;

}

Kısaca CSS kodlarımızı nasıl yazabileceğimizi öğrendikten sonra şimdi CSS seçicileri kullanarak nasıl HTML taglarına erişebiliyoruz onlara bakalım.

## Seçiciler

### **Genel seçici \***

Bu seçiciyi kullanarak tüm etiketlere CSS uygula demiş oluyoruz.

\*{

margin:0;

padding:0;

}

div \*{

color:orange;

}

İlk kullanımda tüm elementlerin margin ve padding değerlerini sıfırlamış olduk. İkinci kullanımda div \* diyerek tüm div elementleri içindeki elementlerin yazı rengine erişmiş olduk.

### **Element Seçiciler (Element Selectors)**

Bu seçiciler ile doğrudan HTML etiket isimlerini kullanarak CSS uygulayabiliriz.

div{

background-color: orange;

}

Tüm div etiketleri etkilenir.

### **Sınıf Seçiciler (Class Selectors)**

Bu seçiciler ile sınıf atadığımız etiketlere CSS uygulayabiliriz. Sınıf ismine erişmek için sınıf isminin başına nokta . ekliyoruz.

.turuncu{

background-color: #FFA500;

}

p.mavi{

color:blue;

}

<p class="turuncu">Arka plan rengim turuncu</p>

<div class="turuncu">Arka plan rengim turuncu</div>

Burada ikinci kullanımda p.mavi sınıf ismi mavi olan p elementlerine erişmiş olduk.

### **Id Seçiciler (Id Selectors)**

Bu seçiciler ile id atadığımız elementlere CSS uygulayabiliriz. Id' ler tek bir elemente ait olmalıdırlar. Id özelliğine erişmek id' nin başına # ekliyoruz.

#mavi{

background-color: #0000FF;

}

#lila{

color: #c8a2c8;

}

<p id="mavi">Arka plan rengim turuncu</p>

<div id="lila">yazı rengim lila</div>

### **Özellik Seçiciler (Attribute Selectors)**

Bu seçiciler ile özelliğini belirttiğimiz elementlere CSS uygulayabiliriz. Özelliğin içi boş olsada element bundan etkilenecektir. Özelliklere erişmek için yapmamız gereken tek şey köşeli parantezler içinde özelliğin ismini [attribute] şeklinde yazıyoruz.

[name]{

color: orange;

}

<button name="">gönder</button>

<ul>

<li name="html">HTML</li>

<li name="css">CSS</li>

</ul>

Bu şekilde name attribute alan tüm elementler etkilenir.

.btn[disabled] {

color: orchid;

}

<button class="btn" disabled="disabled">Submit</button>

Burada sınıfı .btn ve niteliği(attribute) [disabled] olan butona CSS uyguladık.

div[title="deneme"] {

background-color: orange;

}

<div title="Deneme">Lorem, ipsum dolor.</div>

<div title="deneme">Lorem, ipsum dolor.</div>

<div name="denemefalan">Lorem, ipsum.</div>

Burada tam eşleşen özelliğe CSS uyguladık. Büyük-küçük harf duyarlılığı vardır.

div[title~="isim"] {

color: orange;

}

<div title="isim">Lorem, ipsum dolor.</div>

<div title="isimler">Lorem, ipsum dolor.</div>

<div title="isim ve şehirler">Lorem, ipsum dolor.</div>

Burada ~= ifadesi ile title özelliği "isim" içeren divlere eriştik.

a[href ^= "https"] {

color: palegreen;

}

<a href="https://www.google.com/">google</a>

<a href="https://github.com/">github</a>

<a href="http://github.com/">github</a>

Burada ^= ifadesi ile href özelliği "https" ile başlayan a etiketlerine eriştik.

a[href \*= "http"] {

color: palegreen;

}

<a href="https://www.google.com/">google</a>

<a href="https://github.com/">github</a>

<a href="http://github.com/">github</a>

Burada \*= ifadesi ile href özelliği "http" içeren a etiketlerine eriştik.

div[class$="test"] {

background: yellow;

}

<div class="bir\_test">Lorem, ipsum dolor.</div>

<div class="iki-test">Lorem, ipsum dolor.</div>

<div class="uc test">Lorem, ipsum dolor.</div>

<div class="dorttest">Lorem, ipsum dolor.</div>

Burada $= ifadesi ile class özelliği sonunda "test" içeren divlere eriştik.

a[href\*="https"][href$="com"] {

color: orange;

}

<a href="https://www.google.com/"> Google</a>

<a href="https://reactjs.org/">React.js</a>

<a href="https://css-tricks.com/">Css Tricks</a>

Burada https ile başlayan ve sonunda com olan a etiketlerine eriştik.

Artık ana hatlarıyla öğrendiğimiz CSS seçicilere daha yakından bakabiliriz.

### **Grup Seçiciler (Group Selectors)**

Çoğunlukla etiketlere verilen CSS özellikleri benzer veya ortak olabilir. Bu gibi durumlarda seçicileri gruplayarak daha temiz CSS dosyaları oluşturabiliriz.

h1,h2,h3{

color:orange;

}

Tüm h1, h2, h3 etiketlerine erişmiş olduk.

p.turuncu{

color:orange;

}

<p class="turuncu">turuncu yazı</p>

<p>normal yazı</p>

Burada sınıf ismi .turuncu olan p etiketlerine ulaştık.

### **Çocuk seçiciler (Child Selectors)**

Artık etiketleri birbiriyle olan hiyerarşik durumuna göre seçerek CSS özelliklerini belirleyeceğiz. Child selector, kendisi ve kendisini sarmalayan bir üst etiketle olan ilişkiyi gösterir ve > işareti ile ifade edilir.

p > span{

color:orange;

}

p > span >b{

color:blue;

}

div > ul >li#first{

color:red;

}

<p>

<span>child element</span>

</p>

Parent etiketi yani bir üst kapsayıcısı p olan span etiketine ulaştık.

<p>

<span>Burası turuncu renkte <b>mavi renkte yazılacak</b></span>

</p>

<p>

<span>Burası turuncu renkte <b>mavi renkte yazılacak</b></span>

</p>

<div>

<ul>

<li id="first">bir</li>

<li>iki</li>

</ul>

</div>

#### **Torun-Soy Seçiciler (Descentad Selectors)**

Bir kapsayıcı yani parent element altındaki tüm etiketlere ulaşmak için kullanılır. Her ulaşılacak etiket arasına boşluk konulur.

div p{

background-color:blue;

}

Burada div içinde olan tüm p etiketlerine ulaşırız.

<div>

<p>Bu p etiketi arka planı mavi renk</p>

<ul>

<li>

<p>Bu p etiketi arka planı mavi renk</p>

</li>

</ul>

<p>Bu p etiketi arka planı mavi renk</p>

</div>

#### **Genel Kardeş Seçiciler (General Sibling Selectors)**

Aynı parent etikete sahip olan ve birbiri ardına gelen etiketleri seçmek için kullanılır. AltGr+ü Kambinasyonuyla oluşan ~ işareti ile gösterilir.

ul ~ p{

color:orange;

}

<div>

<p>Lorem, ipsum.</p>

<ul>

<li>

<p>Lorem, ipsum dolor.</p>

</li>

</ul>

<h3>Lorem, ipsum dolor.</h3>

<p>Bu p etiketi turuncu renkte</p>

</div>

Burada dikkat edilmesi gereken iki nokta var. Birincisi <p> etiketi <ul> etiketinden sonra gelmeli (arada başka etiketler olabilir) ve ikiside aynı düzeyde yani aynı parent etiketine sahip olmalılar.

#### **Bitişik Kardeş Seçiciler (Adjacent Sibling Selector)**

Genel kardeş seçiciden tek farkı belirtilen etiketler bitişik arka arkaya gelmeliler. + işareti ile gösterilir.

ul + p{

color:green;

}

<div>

<p>Lorem, ipsum.</p>

<ul>

<li>

<p>Lorem, ipsum.</p>

</li>

</ul>

<p>Bu p etiketinin yazı rengi yeşil</p>

</div>

### **Sahte Sınıf Seçiciler (Pseudo classes)**

Sahte sınıflar kullanarak HTML etiketlerine CSS uygulayabiliriz. Kullanımları selector:pseudo class{}şeklindedir.

#### **:link**

Daha önce tarayıcıda açılmamış linke CSS uygular. **Not:** Etkisini görmek için tarayıcıya geçmişi temizlemek gerekebilir.

a#google:link {

color: red;

}

<a id="google" href="https://www.google.com/">Google</a>

#### **:hover**

Seçici ile işaretlenen etiketin üzerine mouse ile gelindiğinde CSS uygalanır ve mouse üzerinde değilken etki kaybolur.

a.test:hover {

color: red;

}

<a class="test" href="https://www.github.com/">Github</a>

#### **:active**

Mouse ile tıklandığında CSS uygulanır. Tıklama kaldırıldığında etki kaybolur.

a:active {

position: relative;

top: 5px;

}

<a href="#">Submit</a>

#### **:first-child**

İlk child etikete CSS uygulanır.

ul > li:first-child {

color: orange;

}

<ul>

<li>lorem</li>

<li>lorem</li>

<li>lorem</li>

<li>lorem</li>

</ul>

#### **:last-child**

Son child etikete CSS uygular.

#### **::first-letter**

Blok seviyesinde bir etiketteki ilk harfe CSS uygular. <span> blok seviyesinde bir etiket olmadığından bu pseudo class uygulanamaz.

p::first-letter {

font-size: 20px;

color: orange;

}

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

#### **::first-line**

Blok seviyesinde bir etiketin ilk satırına CSS uygular.

#### **::before**

Belirtilen etiketin önüne CSS uygular.

p::before {

content: "selam";

color: orange;

}

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

#### **::after**

Belirtilen etiketin sonuna CSS uygular. Kullanımı ::before ile aynıdır.

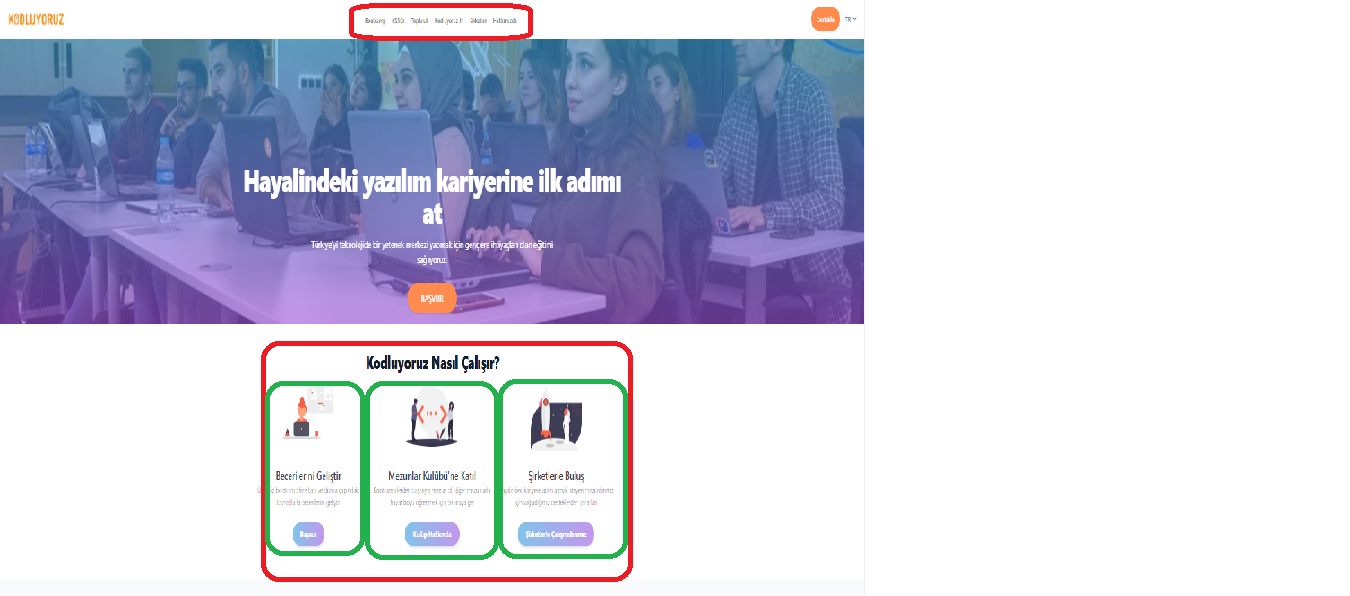
## Kaynaklar

* <https://www.w3schools.com/cssref/css_selectors.asp>
* <https://css-tricks.com/almanac/selectors/>

# CSS Kutu Özellikleri (Margin, Padding, Width, Height) Kullanımı

CSS kutu özelliklerini iyi kavrayabilmek ve kullanabilmek için "kutu model" konusunu incelemek gerekir. Gelin önce "kutu modeli" konusuna bakalım.

## CSS Kutu Modeli



Resimde [Kodluyoruz'un anasayfasını](https://kodluyoruz.org/" \t "_blank) görüyoruz. İşaretlediğim alanlar birer HTML elementleri ve bir yapıyı oluşturan lego parçalarından sadece birkaçı. Kutu denmesinin sebebi de bir bütünü oluşturan lego parçaları veya kutu gibi olması. Bu kutuların aralarında mesafelerin ve içeriklerinin de kendi aralarında bir düzen olduğunu görmekteyiz. Bunun sebebi "kutu özelliklerinin" kullanılıyor olması. Özellikle yeşil kutuların içindeki yazıların belli bir mesafede olmasından ve çizdiğim yeşil hatlara değmemesinden anlayabiliriz.

**Burada kutu modelini şema halinde göstereceğim:**



**Kutu modeli margin,padding,border ve içerikten oluşur.**

* **İçerik(content):** Elementin içinde olan resim, ses ya da yazıdır.
* **Padding:** İçerik ile border arasında olan boşluk.
* **Border:** Padding ile margin'i ayıran sınırdır.
* **Margin:** Kutunun diğer kutularla arasındaki mesafeyi ayarlayan boşluk.

Yukarıda saydığım kavramların dışında kutunun boy ve uzunluğunu belirleyen width ve height kavramları vardır. Bunları da alıştırmalarımızda göstermeye çalışacağım.

## width ve height

width ve height özellikleri ile elementin, yani lego parçamızın boy ve genişliğini ayarlayabiliriz. Bu özellikleri biz yazmadığımız takdirde tarayıcı kendisi ayarlar ve yapacağımız web uygulamasında istenmeyen sonuçlara neden olur :)Bu özellikleri px, em veya % ile kullanabiliriz. % ile kullandığımız zaman elementimiz hangi elementin içindeyse ona göre oranlanır. Yani 800px boyunda bir <div> tag'inin içinde bulunan bir elementin boyunu 50% olarak ayarlarsanız elementin boyu 400px olur.

Max-width özelliği ise bir elementin genişliğini sınırlamak istediğimizde kullanırız. Diyelim ki bir element tarayıcı ekranından fazla geniş veya uzun. **Bu durumda scrollbar(kaydırma çubuğu) çıkar.**

### **Alıştırma-1: Width ve Height Kullanımı**

<style>

p{

width: 0;

height: 0;

}

<style>

<p>

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.

</p>

Bu kodda width ve height'i 0 yaptım. Paragraf tuhaf bir şekilde görünecektir. Bu alıştırmada yapmanız gereken şey paragrafın düzgün görünmesi açısından width ve height'i istediğiniz şekilde kullanın. Burada width ve heigth'i görmeniz için % ve px in ikisini de kullanmanızı tavsiye ediyoruz..

Alıştırmaya aşağıdaki codepen 'den ulaşabilirsiniz.

## margin ve padding

margin

Margin özelliği ile elementin dışında boşluk oluşturmada kullanıldığını öğrenmiştik. Şimdi nasıl kullanıldığını açıklamaya çalışacağız.

margin özelliğini 50px yaptığınızda elementin dört yanında 50px boşluk oluşturulur. Element sağa doğru kayar. Negatif değerler de kullanılabilir. margin değerine -50px yazdığınız takdirde element sola doğru kayacaktır. Dört yanında değil de sadece bir yönde boşluk oluşturmak isteyebilirsiniz. Bunun için de özellikler vardır.

* margin-left: Soldan boşluk bırakır.
* margin-top: Yukarıdan boşluk bırakır.
* margin-bottom: Aşağıdan boşluk bırakır.
* margin-right: Sağdan boşluk bırakır.

Bu değerleri tek bir satırda ayarlamanız mümkündür.

margin: 100px 150px 60px 50px;

Bir elementi ortalamak için margin:auto değeri vermelisiniz. Bunun düzgün çalışması için de width özelliğinin de kullanılmasını öneririm. Böylelikle sağdan ve soldan eşit olarak ortalanmış olur. Aksi halde boyutu belli olmayan bir kutuyu ortalamayabilir.

### **padding**

Şimdi gelelim padding özelliğine. Elementin içindeki yazıların **çerçeve**(border) ile mesafesini padding ile ayarlarız. margin özelliğinde olduğu gibi padding:50px yaptığımızda içeriğin her tarafı için 50px boşluk bırakıyoruz. Belirli bir tarafta padding bırakmak için;

* padding-top: içeriğin yukarısında boşluk bırakır.
* padding-bottom: içeriğin aşağısında boşluk bırakır.
* padding-left: içeriğin solunda boşluk bırakır.
* padding-right: içeriğin sağında boşluk bırakır.

Bunları tek tek tanımlamak yerine tek bir satırda belirlemek mümkündür.

padding: 10px 20px 30px 40px;

/\* Burada üst boşluk 10px,sağ boşluk 20px,alt boşluk 30px ve sol boşluk 40px dir.\*/

padding: 10px 20px 30px;

/\* Üst boşluk 10px,sağ ve sol boşluk 20px,alt boşluk 30px dir. \*/

padding: 10px 20px;

/\* üst ve alt boşluk 10px,sağ ve sol boşluk 20px \*/

Şimdi ikinci alıştırmaya geçelim.

### **Alıştırma 2-Kutu Özelliklerinin Kullanımı**

Alıştırmaya aşağıdaki Codepen'den ulaşabilirsiniz.

## Kaynaklar

* <https://css.sitesi.web.tr/css-kutu-modeli.html>
* <https://www.yazilimkodlama.com/web/css-kutu-modeli-margin-border-padding-content>
* <http://uzmanimakademi.net/css/css-kutu-modeli>

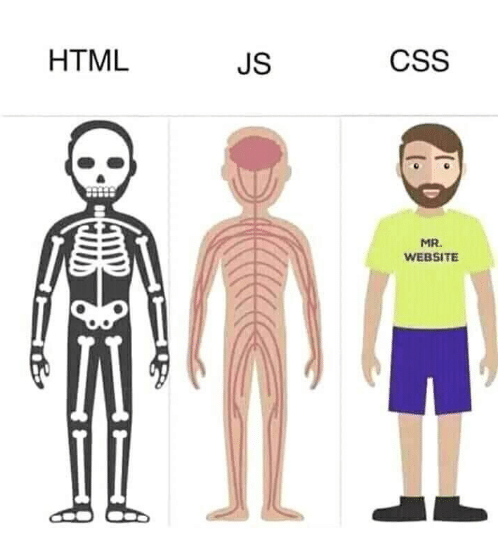
# CSS Özet Çalışması ve Kendimi CSS Konusunda Nasıl Geliştirebilirim

CSS konusunda başka örnekler görmek isterseniz [w3schools](https://www.w3schools.com/css/default.asp) adlı platformu da ziyaret etmenizi öneririz. Bu site üzerinden hem CSS'in alt başlıklarına dahil konuları öğrenebilir hem de başka alıştırmalar yapabilirsiniz.

Daha fazla gelişmek ve kendime bu konuda yetkinlik kazanmak istiyorum diyorsanız, sevdiğiniz tasarımları klonlamayı deneyebilirsiniz böylelikle uygulama yaparak kendinizi daha çok geliştirebilirsiniz. Sonrasında tasarımları modifiye edebilir ve daha özgün tasarımlarla web sitesi hazırlayabilirsiniz. Bizim ödevlerimizde de sık sık vurguladığımız gibi tarayıcıların **inspect** yani **incele** özelliğini bol bol kullanın. Dolaştığınız sitelerde bazı komutları kapatıp açarak ne işe yaradıklarını pratik edebilirsiniz.

Sonuç olarak her şey alıştırma yapmaktan ve bilgilerimizi diri tutmakta bitiyor. :)

## Sayfamıza Biraz Makyaj Yapalım

[](https://raw.githubusercontent.com/Kodluyoruz/taskforce/main/css/odev1/figures/htmlcssjs.png)

Herkese merhaba arkadaşlar kanalımıza hoş geldiniz, bugün HTML sayfamıza hafif bir makyaj yapacağız. Komik gelse de aslında tam olarak böyle yapacağız. Bir HTML sayfası oluşturup buna CSS ile tasarımsal açıdan güzellikler katacağız, HTML sayfamıza güzelliği getireceğiz.

* HTML sayfasını sıfırdan oluşturacaksınız. Eklemek istediğiniz ekstra özellikler tamamen size kalmış durumda.
* Sitemiz birkaç sayfadan oluşacak. Yani menü kısmında linkler vererek başka sayfaya geçilebilecek.
* Renkler tamamen sizin zevkinize kalmış durumda. Fakat renkler konusunda biraz yardım almak isterseniz [Colorhunt](https://colorhunt.co/" \t "_blank) renk paletleri konusunda muazzam bir site.
* Yazacağınız CSS etiketlerini Inline ve Internal yazabilirsiniz. External kullanmak tamamen sizin tercihinizdir.
* CSS yazarken noktalı virgülleri unutmayın. Biliyorum unutacaksınız, olur böyle şeyler...
* CSS ile ilgili yardımcı kaynak için [w3schools.com](https://www.w3schools.com/w3css/defaulT.asp)'un CSS tutorialını, Türkçe kaynak için [Fatih Hayrioğlu](https://fatihhayrioglu.com/)'nun web sitesini kullanabilirsiniz.
* Kendini tekrar eden yapılar kullanmamaya özen gösteriniz. Ya da kendini tekrar eden yapılarda kullandığımız özelliği kullanın. (İpucu: [Inline(Etikete Özel), Internal(Aynı Dosyada) ve External(CSS Dosyasında) CSS Kullanımı](https://www.youtube.com/watch?v=GiTOThmpHbk&list=PLGrTHqyRDvx501K3-IMgS1fz-KfEB37gM&index=6" \t "_blank))
* Sayfalarınızda kullandığınız fontlar için daha önce de videolarda bahsettiğimiz [Google Fonts](https://fonts.google.com/)'u kullanabilirsiniz.
* Ana sayfada bulunan listelerin noktalarını ortalamak için list-style-position: inside'i kullanabilirsiniz.

[](https://raw.githubusercontent.com/Kodluyoruz/taskforce/main/css/odev1/figures/webpage.gif)

Bu ödevde sizlerden istediğimiz hayal gücünüzü kullanarak öğrendiğiniz ve araştırarak bulduklarınızla birlikte güzel bir site oluşturmanız. Burada oluşturduğumuz sitenin tasarımı hoşunuza gitmese dahi burada önemli olan nokta CSS elemanlarını efektif bir şekilde kullanabiliyor muyuz onu görmek. Tasarımı ilerleyen ödevlerde daha güzelleştireceğiz.

Hepinize başarılar, kolay gelsin.

# Ödev 2

## Google Ana Sayfasını Tasarlamak

Hepimiz her gün Google kullanıyoruz ve çok işimize yarıyor değil mi? Her gün Google'da milyonlarca arama yapılıyor ve hatta siz de bu sayfaya gelmek için Google'ı kullanmış olabilirsiniz. Peki Google'ın geçmişten günümüze nasıl geliştiğini hiç merak ettiniz mi?

[Google](https://tr.wikipedia.org/wiki/Google) 1996 yılında kuruldu ve ilk versiyonunu 1998 yılında yayınladı. 1998, çok uzun bir süre önce değil mi? İlk versiyonu ile şu anki versiyonu arasında büyük fark var tabii ki. Peki size Google'ın ilk versiyonunu gösterebileceğimizi söylesek ne hissederdiniz?

İnternetteki gelmiş geçmiş bütün web sitelerini görebileceğiniz Wayback Machine adında bir web arşivi bulunmakta. [Google 1998](https://web.archive.org/web/19981202230410if_/http:/www.google.com/) linkinden Google'ın ilk versiyonu nasılmış görebilirsiniz. Oldukça garip öyle değil mi? Garip olmasının yanında bu sizin HTML bölümündeki üçüncü ve son ödeviniz olacak. Bu sayfayı tasarlamanızı istiyoruz.

* Bu sayfada şu ana kadar öğrendiğiniz her şeyi kullanabilirsiniz. Bu sizin HTML becerilerinizi oldukça iyi bir şekilde geliştirmenizi sağlayacaktır.



* Butonların çalışmaması hiç sorun değil. Sadece tasarımsal olarak bu görüntüye benzesin ve aşağıdaki linkler çalışıyor olsun yeterli.
* Tasarladığınız bölümler ile alakalı kodunuzda açıklama satırlarına yer veriniz.
* Sayfa ile alakalı detaylara sayfanın üzerine sağ tıklayıp "İncele/Inspect" diyerek ulaşabilirsiniz.
* Bu [logo](https://web.archive.org/web/19990504112211im_/http:/www.google.com/google.jpg" \t "_blank)'yu kullanabilirsiniz.

İleride göreceğimiz CSS'den sonra Google'ın bugünkü halini de tasarlayacağız. Ama öncelikle bakalım Larry Page ve Sergey Brin başlangıçta nasıl yapmışlar!

Hepinize başarılar ve kolay gelsin!

## Kaynaklar

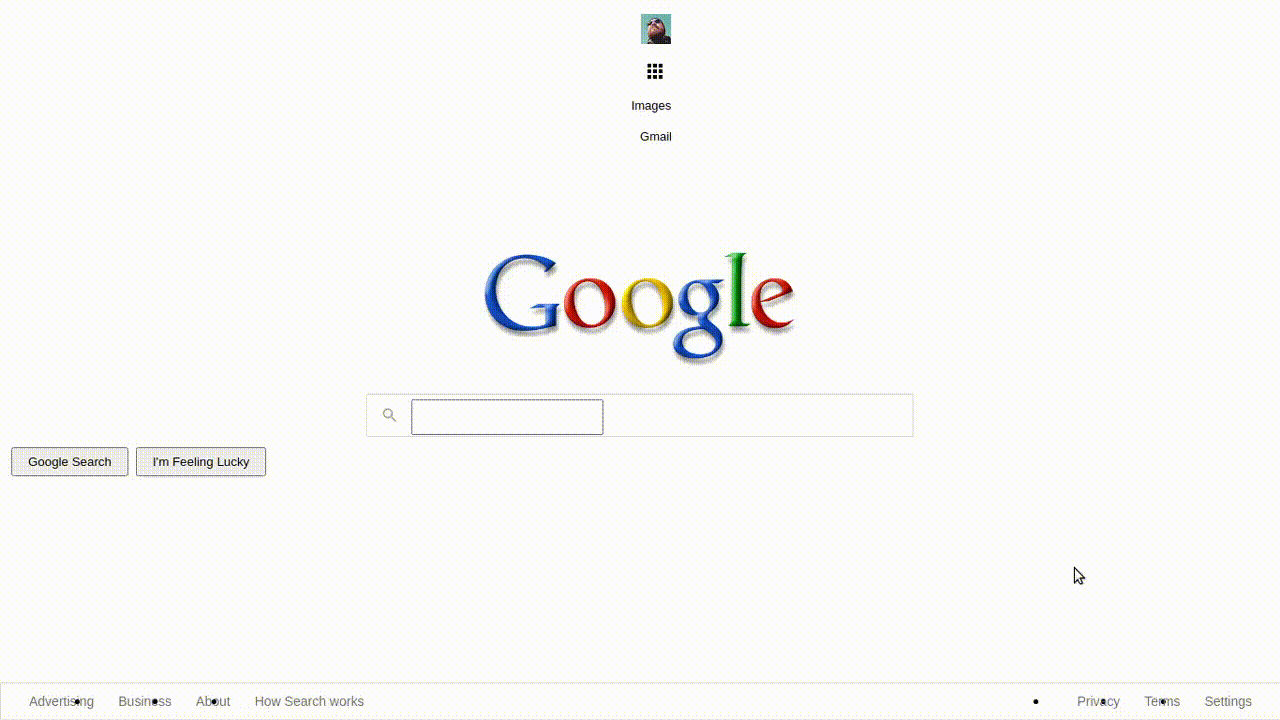
* <https://tr.wikipedia.org/wiki/Google>
* <https://web.archive.org/>

# Ödev 3

## Google Ana Sayfasını Tasarlamak

Bir zamanlar basit ama gururlu bir Google Ana Sayfası vardı, hatırladınız mı? Yıllaar yıllar geçti ve o ana sayfa gelişti, serpildi ve bugünkü halini aldı. HTML'in son ödevinde Google'ın ilk tasarımı üzerinde çalışmıştık ve CSS bölümünde günümüzdeki halini tasarlayacağımızı söylemiştik. Evet, o gün bugündür arkadaşlar. Bu ödevde Google'ın bugünkü ana sayfasını tasarlayacağız.

Sizler için HTML yapısını ve basit CSS'i hazırladık. Sizlerden istediğimiz orijinali inceleyip detayları işlemeniz.



[Google Homepage Projesi](https://github.com/Kodluyoruz/taskforce/tree/main/css/odev2/google_homepage)'ne buradan erişebilirsiniz.

Projeyi indirdikten ya da cloneladıktan sonra Visual Studio Code programında LiveServer eklentisi ile açmanızı öneririz.

[Google Ana Sayfa](https://web.archive.org/web/20191130234759if_/https:/www.google.com/) için arşiv linkini kullanabilirsiniz ya da [Google](https://www.google.com/)'ın şimdiki ana sayfasını kullanabilirsiniz.

Bu ödevi yapabilmeniz için bolca "İncele/Inspect"'ten yararlanmanız gerekecektir. Sayfadayken sayfaya sağ tıklayıp "İncele/Inspect" demeniz yeterlidir.

Fark ettiyseniz logo eski bir Google logosu. Bu logoyu günümüzde logo ile değiştirmelisiniz. Kullanmanız gereken logo assets klasöründe bulunmakta.

Üstteki alanı sağ tarafa alın ve fotoğrafı kendi fotoğrafınız ile değiştirin ve fotoğrafın kenarlarını yarıçap özelliği ile düzenleyiniz.

Arama yapılacak alanın kenarlarını yarıçap özelliği ile düzenleyiniz.

Arama simgesi gibi sesle arama simgesini de siz ekleyin. (Nasıl yapıldığını görmek için Google Ana Sayfa'dan İncele ile yazılanları inceleyiniz.)

Arama alanında yazı yazılan yerin genişliği 480px olmalı ve kenarlığı olmamalı.

Buttonları ortaya alıp üstünden ve sağından boşluklar veriniz. Button kenarları 1px kalın #f2f2f2 renginde olmalı. Yazı tipi Arial, yazı rengi #5f6368, yazı boyutu 14px olmalı. Button yüksekliği 36px olmalı.

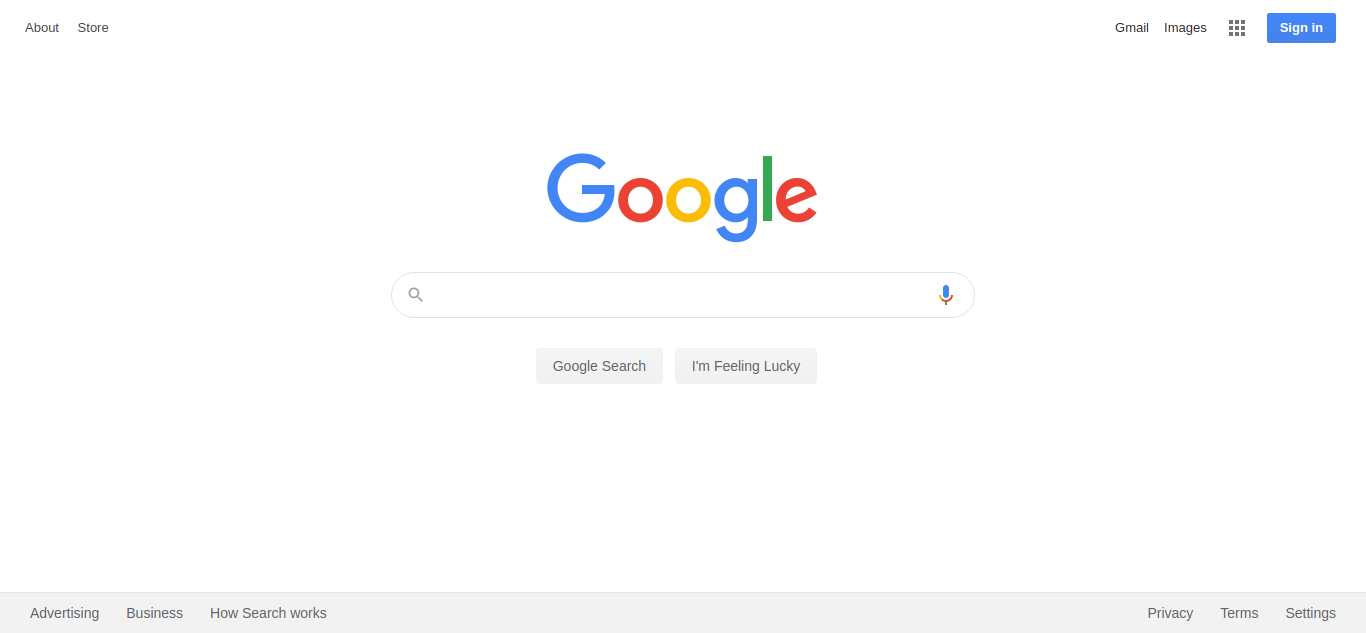
Buttonlara aynı arama alanında olduğu gibi gölge veriniz.

Footerda arkaplan rengini #f2f2f2 yapınız ve liste noktalarını ortadan kaldırın. (Bir önceki ödevde söylediğimiz [w3schools.com](https://www.w3schools.com/w3css/defaulT.asp)'un CSS tutorialını, Türkçe kaynak için [Fatih Hayrioğlu](https://fatihhayrioglu.com/)'un sitesini kullanabilirsiniz.)

CSS dosyasında kullanılan bütün elementleri araştırıp ne işe yaradıklarını öğrenin. Daha sonra bunları kullanabileceğiniz projelerinizde kullanmaya özen gösterin.

Tasarımı mümkün olduğunca benzetmeniz gerekmekte. Buttonların, arama kısmının çalışmaması önemli değil.

Kod yazarken yorum satırları kullanmaya özen gösteriniz. ödevin indirilebilir içeriklerine [buradan](https://drive.google.com/drive/folders/1Judez8Dqey-BHDjxZ1PnUwGhIMERNkbV?usp=sharing) ulaşabilirsiniz.



Bu ödev ile birlikte CSS bölümünü tamamladınız, tebrik ederiz!

Hepinize başarılar ve kolay gelsin!😄