

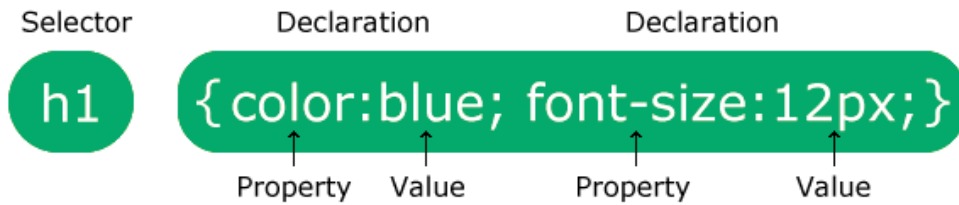
CSS

1 CSS Nedir?

CSS (Cascading Style Sheets), web sayfalarının nasıl görüneceğini tarif eden bir stil şablonu dilidir. HTML ile yapılan yapısal işleri (yani bir web sayfasının iskeletini) stilize etmek ve görsel düzenini kontrol etmek için kullanılır. CSS, web geliştirmede önemli bir role sahiptir çünkü bir web sayfasının layout'unu (düzeni), renklerini, fontlarını ve hatta animasyonlarını kontrol edebilir. Temel olarak CSS ile bir web sayfasının sunumunu ve kullanıcı deneyimini iyileştirebilirsiniz.

CSS, web'in ilk günlerinde web sayfalarının görünümünü standart hale getirmek ve HTML'den sunumu ayırmak amacıyla 1996 yılında W3C (World Wide Web Consortium) tarafından geliştirilmiştir. O günden bu yana CSS sürekli olarak gelişmiş ve yeni özellikler eklenmiştir. CSS3 şu anda en yaygın kullanılan versiyondur.

CSS, tarayıcıya HTML elemanlarının nasıl görüntüleneceğini söyler. CSS kuralları genellikle bir seçici ve bir veya daha fazla özellik-değer çiftinden oluşur. Seçici, stilin uygulanacağı HTML elementini belirtir. Özellikler, değiştirilecek stilin türünü (örneğin, renk, genişlik, yükseklik) belirtirken değerler bu özelliklere verilen spesifik değerlerdir.



CSS, web tasarımında kritik bir öneme sahiptir. İyi tasarlanmış bir CSS dosyası web sayfalarının daha hızlı yüklenmesini, cihazlar ve tarayıcılar arasında tutarlı bir görünüm sağlamasını ve genel kullanıcı deneyimini iyileştirmesini sağlar. Ayrıca içerik ve sunumu birbirinden ayırarak web sitelerinin bakımını ve güncellenmesini kolaylaştırır.

1.1 CSS ile Yapılabilecekler

Görsel Stiller: Metin renkleri, fontlar, arka planlar gibi görsel stilleri tanımlayabilirsiniz.

Layout: Sayfa düzenini kontrol edebilir, sütunlar oluşturabilir ve elementleri istediğiniz gibi konumlandırabilirsiniz.

Responsive Tasarım: Ekran boyutuna göre farklı stil kuralları uygulayarak mobil cihazlar da dahil olmak üzere tüm cihazlarda iyi görünen siteler oluşturabilirsiniz.

Animasyonlar ve Geçişler: Sayfalarınıza canlılık katmak için animasyonlar ve geçiş efektleri ekleyebilirsiniz.

CSS modern web geliştirmenin vazgeçilmez bir parçasıdır ve etkili bir şekilde kullanıldığında kullanıcıların web deneyimini büyük ölçüde iyileştirebilir.

2 CSS Seçicileri

CSS seçicileri HTML belgesindeki öğelere stil uygulamak için kullanılır. Çeşitli seçici türleri farklı öğe gruplarını hedeflemenize olanak tanır.

2.1 Element Seçicisi

Element seçicisi belirli bir HTML elementine stil uygulamak için kullanılır.

Örnek

```
p {  
  color: red;  
  font-size: 16px;  
}
```

Bu CSS kodu, tüm paragraf metinlerini kırmızı renkte ve 16 piksel font boyutunda gösterir.

2.2 ID Seçicisi

ID seçicisi id özelliğine sahip belirli bir HTML elementine stil uygulamak için kullanılır. Her ID benzersiz olmalıdır.

Örnek

```
#baslik {  
  color: red;  
  font-size: 16px;  
}
```

Bu CSS kodu ID'si baslik olan elementin metinlerini kırmızı renkte ve 16 piksel font boyutunda gösterir.

2.3 Sınıf Seçicisi

Sınıf seçicisi class özniteliğine sahip bir veya daha fazla HTML elementine stil uygulamak için kullanılır.

Örnek

```
.highlight {  
  font-weight: bold;  
  color: blue;  
}
```

Bu kod "highlight" sınıfına sahip tüm elementlerin metin rengini mavi yapar ve fontu kalın yapar.

2.4 Gruplandırma Seçicisi

Birden fazla seçiciyi virgül ile ayırarak gruplayabilir ve aynı stil kurallarını birden fazla seçiciye uygulayabilirsiniz.

Örnek

```
h1, h2, p {  
  color: green;  
}
```

Bu kod <h1>, <h2> ve <p> elementlerinin metin rengini yeşil yapar.

2.5 Alt Seçici

Alt seçici belirli bir elementin altındaki tüm öğeleri (doğrudan çocukları ve daha alt seviyedeki tüm öğeleri) hedef alan seçicidir. İki seçici arasına boşluk koyarak kullanılır.

Örnek

```
div p {  
  color: green;  
}
```

Bu kod div elementlerinin içindeki tüm p (paragraf) elementlerini hedef alır ve onların metin rengini yeşil yapar. Burada div'in doğrudan çocukları olan p'lerin yanı sıra div içinde daha derin seviyelerde bulunan p'ler de hedef alınır.

2.6 Child Selector (Çocuk Seçici (>))

Çocuk seçici belirli bir elementin doğrudan çocuklarını hedef almak için kullanılır. Yani bir elementin altında bulunan ancak daha derin seviyedeki torunları değil sadece bir sonraki seviyedeki çocukları kapsar.

Örnek

```
ul > li {  
  color: red;  
}
```

Bu örnek bir ul elementinin doğrudan çocukları olan li elementlerini hedef alır ve onların metin rengini kırmızı yapar.

2.7 Pseudo Sınıflar

Pseudo-sınıflar CSS'te özel durumları tanımlamak için kullanılan seçicilerdir. Bir elementin belirli bir durumda olduğunu ifade ederler; örneğin bir kullanıcının bir bağlantının üzerine geldiği an veya bir form elemanın odaklanılmış (fokuslanmış) olduğu durum. Pseudo-sınıflar element seçicilerine ek olarak : sembolü ile kullanılır ve bu durumlar CSS ile stil

verilmeden önce doğrudan HTML tarafından tanımlanamaz. Böylece dinamik kullanıcı etkileşimlerine yanıt olarak stil değişiklikleri uygulamak mümkün hale gelir.

:hover

Kullanıcı bir elementin üzerine fareyle geldiğinde uygulanır. Genellikle bağlantılar ve düğmeler üzerinde görsel geri bildirim sağlamak için kullanılır.

Örnek

```
a:hover {  
    color: red;  
}
```

Bu kod web sayfasındaki herhangi bir linkin renginin Mouse ile üzerine gelince kırmızıya dönüşmesini sağlar.

:focus

Element klavye ile erişim (örneğin, tab tuşu ile gezinme) veya fare tıklaması yoluyla odaklandığında uygulanır. Genellikle Form elemanlarının kullanıcı tarafından seçildiğini göstermek için kullanılır.

Örnek

```
input:focus {  
    border-color: blue;  
}
```

Bu kod web sayfasındaki herhangi bir input elementi odaklandığında kenarlık rengini mavi yapar.

:active

Element üzerinde bir tıklama işlemi gerçekleştiği sürece uygulanır. Kullanıcı tarafından bir elemente tıklanıp tıklanmadığını görsel olarak belirtmek için kullanılır.

Örnek

```
a:active {  
    color: yellow;  
}
```

Bu kod web sayfasındaki herhangi bir linke tıklandığında linkin rengini sarı yapar.

:visited

Kullanıcı tarafından daha önce ziyaret edilmiş bir bağlantıya uygulanır. Ziyaret edilmiş bağlantıların stilini değiştirmek için kullanılır.

Örnek

```
a:visited {  
    color: purple;  
}
```

Bu kod web sayfasındaki herhangi bir linke tıklandıktan sonra linkin rengini mor yapar.

:first-child ve :last-child

Bir elementin ebeveyn element içindeki ilk veya son çocuk olması durumunda uygulanır. Liste öğeleri veya paragraflar gibi gruplanmış elementler üzerinde kullanılabilir.

Örnek

```
li:first-child {  
    font-weight: bold;  
}
```

Bu kod bir web sayfasındaki listelerde bulunan ilk li elementlerinin yazı tipini kalın yapar.

:nth-child(n)

Ebeveyninin n'inci çocuğu olan elementleri hedef alır. Bu pseudo-sınıf karmaşık dizileri seçmek için kullanılabilir.

Örnek

```
li:nth-child(3) {  
    background-color: gray;  
}
```

Bu kod web sayfasındaki listelerde bulunan li elementlerinin 3.lerinin arkaplan rengini gri yapar.

```
li:nth-child(odd) {  
    background-color: gray;  
}
```

Bu kod web sayfasındaki listelerde bulunan li elementlerinden (çift sayı) 2'nin katı olanlarının arkaplan rengini gri yapar. Tek olanlar için even yazılır.

:not(selector)

Belirtilen seçiciye uymayan elementler için stil tanımlar. Örneğin belirli bir sınıfa sahip olmayan tüm elementleri hedef almak için kullanılabilir.

Örnek

```
div:not(.special) {  
  color: green;  
}
```

Bu kod sınıfı special olmayan tüm div elementlerinin yazı rengini yeşil yapar.

2.8 Pseudo Elementler

Pseudo-elementler CSS ile bir HTML belgesinde belirli bir elementin belirli bölümlerine stil uygulamak için kullanılan özel seçicilerdir. Bu seçiciler mevcut HTML yapısını değiştirmeden öğelerin belirli kısımlarına erişmenizi ve bunları stilize etmenizi sağlar. Pseudo-elementler içerik eklemek, belirli bir bölümü özelleştirmek veya dekoratif efektler yaratmak için yaygın olarak kullanılır.

Pseudo-elementler iki nokta (::) ile başlar ve CSS2'ye kadar tek nokta (:) ile kullanımları yaygındı. CSS3 ile pseudo-sınıflarla aralarındaki farkı daha net hale getirmek için pseudo-elementler için iki nokta (::) notasyonu standart hale getirildi.

::before ve ::after

Bu pseudo-elementler seçili elementin içeriğinden önce veya sonra içerik eklemek için kullanılır. content özelliği bu pseudo-elementlerle birlikte kullanılmak zorundadır.

Örnek

```
p::before {  
  content: "Başlangıç: ";  
  color: red;  
}  
  
p::after {  
  content: " Son";  
  color: blue;  
}
```

Bu örnek, her <p> elementinin başına kırmızı renkli "Başlangıç: " metnini ve sonuna mavi renkli " Son" metnini ekler.

::first-letter ve ::first-line

::first-letter pseudo-elementi bir blok seviyesi elementinin (örneğin, bir paragraf) ilk harfine stil uygulamak için kullanılır. ::first-line ise aynı elementin ilk satırına stil uygular.

Örnek

```
p::first-letter {  
  font-size: 200%;  
  font-weight: bold;  
}  
  
p::first-line {  
  color: green;  
}
```

Bu örnek her <p> elementinin ilk harfini büyütür ve kalın yapar; aynı zamanda ilk satırın rengini yeşil yapar.

::selection

::selection pseudo-elementi kullanıcının bir metni seçtiği (örneğin, fare ile sürükleyerek) zaman seçilen metin için stil uygulamak için kullanılır.

Örnek

```
::selection {  
  color: white;  
  background-color: black;  
}
```

Bu örnek seçilen metnin rengini beyaz ve arka planını siyah yapar.

3 CSS Kodunu Web Sitesine Ekleme

CSS kodunu bir web sitesine eklemenin üç temel yolu vardır: Harici CSS dosyaları kullanmak, dahili CSS kullanmak ve satır içi CSS kullanmak. Her bir yöntemin kendine özgü kullanım durumları ve avantajları vardır.

3.1 Harici CSS Dosyaları Kullanmak

Harici CSS stil kurallarınızı ayrı bir .css dosyasında tutmanızı sağlar. Bu yöntem stil bilgilerini HTML dosyanızdan ayırmanızı ve aynı CSS dosyasını birden fazla sayfada yeniden kullanmanızı mümkün kılar. Bu yöntem büyük projeler için idealdir çünkü sitenizin tutarlılığını korumayı kolaylaştırır ve stil değişikliklerini yapmayı basitleştirir.

Nasıl yapılır?

Öncelikle kullandığınız editör ile yeni bir css dosya oluşturunuz. İçerisine web sitesine dahil etmek istediğiniz css kodlarınızı yazınız. Ardından HTML dosyanızda <head> bölümü içinde bu CSS dosyasını <link> etiketi ile referanslayınız.

Örnek

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="styles.css">
</head>
<body>

  <h1>Merhaba Dünya!</h1>
  <p>Bu bir paragraftır.</p>

</body>
</html>
```

Bu örnekte styles.css isimli css dosyası html sayfasının head etiketleri arasında kullanılan link etiketiyle eklenmiştir.

3.2 Dahili CSS Kullanmak

Dahili CSS <style> etiketi kullanılarak HTML dosyanızın <head> bölümüne doğrudan yazılır. Bu yöntem sayfa özelinde stil değişiklikleri yapmanız gerektiğinde kullanışlıdır ancak sitenizin geri kalanıyla stil tutarlılığını korumak istiyorsanız harici CSS kullanmak daha iyidir.

Örnek

```
<!DOCTYPE html>
<html>
<head>
  <style>
    body {
      background-color: lightblue;
    }
    h1 {
      color: navy;
      margin-left: 20px;
    }
  </style>
</head>
<body>

  <h1>Merhaba Dünya!</h1>
  <p>Bu bir paragraftır.</p>

</body>
</html>
```

Bu örnekte sayfa için yazılan css kodları style etiketleri içerisine yazılarak sayfaya dahil edilmiştir.

3.3 Satır İçi CSS Kullanmak

Satır içi CSS stil kurallarını doğrudan HTML elementlerinin içine style özniteliği aracılığıyla eklemenizi sağlar. Bu yöntem çok spesifik ve tek seferlik stil değişiklikleri için kullanışlıdır ancak genel olarak kaçınılması gereken bir uygulamadır. Çünkü HTML ile stil bilgilerini karıştırır ve kodun okunabilirliğini azaltır.

Örnek

```
<!DOCTYPE html>
<html>
<body>

  <h1 style="color: navy; margin-left: 20px;">Merhaba Dünya!</h1>
  <p style="background-color: lightblue;">Bu bir paragraftır.</p>

</body>
</html>
```

Her bir CSS ekleme yöntemi belirli durumlar için uygun olabilir. Genel olarak sitenin genel stilini yönetmek için harici CSS dosyaları kullanmak en iyi uygulamadır. Dahili CSS sayfa özelindeki değişiklikler için kullanılabilir. Satır içi CSS ise özel durumlar dışında mümkün olduğunca az kullanılmalıdır.

4 Özellikler ve Değerler

4.1 Renkler

CSS'de renkler web sayfalarına görsel zenginlik ve derinlik katmanın temel yollarından biridir. Renkler kullanıcı deneyimini doğrudan etkileyebilir ve markanızın görsel kimliğini yansıtır.

CSS'de renkleri belirtmek için çeşitli yöntemler kullanılabilir:

İsimlerle Renk Belirleme

CSS, red, blue, green gibi birçok standart renk adını tanır. Bu tanımlamaları doğrudan kullanabilirsiniz. Çoğu tarayıcı tarafından desteklenen renk isimleri için lütfen [tıklayınız](#).

Örnek

```
p {
  color: red;
}
```

Bu kod web sayfasındaki tüm p etiketlerinin yazı rengini kırmızı yapar.

RGB ve RGBA ile Renk Belirleme

RGB ve RGBA, CSS'de renkleri belirtmek için kullanılan iki yaygın yöntemdir. Bu yöntemler kırmızı (Red), yeşil (Green) ve mavi (Blue) renk bileşenlerinin karışımıyla renkleri tanımlarlar. RGBA, RGB'ye ek olarak bir alfa kanalı (saydamlık) bileşeni de sunar.

RGB, renklerin kırmızı, yeşil ve mavi bileşenlerinin 0 ile 255 arasında bir değer alarak karışımıyla oluşturulduğu bir modeldir. Bu üç renk bileşeni milyonlarca farklı renk oluşturmak için bir araya getirilebilir. Sözdizimi `rgb(red, green, blue)` şeklindedir. `red`, `green`, `blue` değerleri her bir renk bileşeninin yoğunluğunu belirtir ve 0 ile 255 arasında bir sayı alır.

Örnek

```
p {  
  color: rgb(255, 0, 0); /* Kırmızı metin rengi */  
  background-color: rgb(0, 255, 0); /* Yeşil arka plan rengi */  
}
```

Bu örnekte, `p` etiketinin metni kırmızı (255,0,0) ve arka planı yeşil (0,255,0) olarak belirlenmiştir.

RGBA, RGB modeline ek olarak bir alfa (alpha) kanalı ekler. Alfa kanalı renklerin saydamlığını kontrol etmek için kullanılır. Değeri 0.0 (tamamen saydam) ile 1.0 (tamamen opak) arasında bir sayı olabilir.

Örnek

```
p {  
  color: rgba(255, 0, 0, 0.5); /* Yarı saydam kırmızı metin rengi */  
  background-color: rgba(0, 0, 255, 0.3); /* Çok saydam mavi arka plan rengi */  
}
```

Bu örnekte, `p` etiketinin metni yarı saydam kırmızı (255,0,0,0.5) ve arka planı çok saydam mavi (0,0,255,0.3) olarak ayarlanmıştır.

Hexadecimal Kodlarla Renk Belirleme

Hexadecimal (altılık veya hex) renk kodları web tasarımında renkleri tanımlamak için sıkça kullanılan bir yöntemdir. Hex kodları genellikle bir diyez işareti (#) ile başlar ve ardından altı haneli bir sayı ve/veya harf dizisi gelir. Bu kodlama sistemi, kırmızı (Red), yeşil (Green) ve mavi (Blue) renk bileşenlerinin yoğunluklarını 00 ile FF arasında bir değerle ifade eder. Bu şekilde 16 milyondan fazla (256^3) farklı renk oluşturulabilir.

Hex renk kodları üç temel renk bileşeninden oluşur:

- İlk iki hane kırmızı (R) bileşenini,
- Sonraki iki hane yeşil (G) bileşenini,
- Son iki hane mavi (B) bileşenini temsil eder.

Her bir bileşen için 00 en düşük yoğunluğu (renk yok) ve FF en yüksek yoğunluğu (tam yoğunluk) ifade eder.

Örnek:

- #000000 tamamen siyah,
- #FFFFFF tamamen beyaz,
- #FF0000 tamamen kırmızı,
- #00FF00 tamamen yeşil,
- #0000FF tamamen mavi

Bazen, her renk bileşeni için aynı değer tekrarlandığında, hexadecimal kodlar üç haneli olarak kısaltılabilir.

Örnek:

- #000 siyah,
- #FFF beyaz,
- #F00 kırmızı,
- #0F0 yeşil,
- #00F mavi

Örnek

```
body {  
  background-color: #3498db; /* Açık mavi arka plan */  
}  
  
h1 {  
  color: #2ecc71; /* Açık yeşil metin rengi */  
}
```

Bu örnekte body elementi için açık mavi bir arka plan rengi ve h1 başlıkları için açık yeşil bir metin rengi belirlenmiştir.

4.2 Arkaplan

CSS'de arka plan ayarlamak web sayfasının görsel çekiciliğini artırmanın temel yollarından biridir. CSS, arka plan rengi, arka plan resmi, arka plan pozisyonu, boyutu ve tekrarlamasını kontrol etmek için çeşitli özellikler sunar.

Arkaplan Rengi Ayarlama (background-color)

Arka plan rengini ayarlamak için background-color özelliği kullanılır. Bu özellik, bir elementin arka planına düz bir renk eklemenizi sağlar.

Örnek

```
body {  
  background-color: #3498db; /* Açık mavi arka plan rengi */  
}
```

Arkaplan Resmi Ekleme (background-image)

Bir elementin arka planına resim eklemek için background-image özelliği kullanılır. Resmin URL'si url() fonksiyonu içinde belirtilir.

Örnek

```
body {  
  background-image: url('background.jpg');  
}
```

Arkaplan Konumunu Ayarlama (background-position)

Arka plan resminin konumunu ayarlamak için background-position özelliği kullanılır. Bu özellik resmin hangi noktada başlayacağını belirlemenize olanak tanır.

Örnek

```
body {  
  background-image: url('background.jpg');  
  background-position: center top; /* Resmi üstte ve ortada konumlandır */  
}
```

Arkaplan Tekrarını Ayarlama (background-repeat)

background-repeat özelliği, arka plan resminin nasıl tekrarlanacağını (veya tekrarlanmayacağını) belirler. Değerler arasında repeat, repeat-x, repeat-y, ve no-repeat bulunur.

Örnek

```
body {  
  background-image: url('background.jpg');  
  background-repeat: no-repeat; /* Resmi tekrarlatma */  
}
```

repeat-x resmi x eksenini boyunca tekrar eder. repeat-y y eksenini boyunca tekrar eder. repeat hem x hem y eksenini boyunca tekrar eder. no-repeat tekrar etmesini engeller.

Arkaplan Kısa Kullanım

Örnek

```
body {  
  background-color: #ffffff;  
  background-image: url("img_tree.png");  
  background-repeat: no-repeat;  
  background-position: right top;  
}
```

Yukarıdaki kodu aşağıdaki gibi tek satırda yazabilirsiniz.

```
body {  
  background: #ffffff url("img_tree.png") no-repeat right top;  
}
```

4.3 Borders (Çerçeveler)

HTML elemanlarının sınırlarını (borders) şekillendirmek, web sayfası tasarımının önemli bir parçasıdır. Border özellikleri kullanılarak elemanların etrafına kenarlık eklenebilir ve bu kenarlıkların kalınlığı, stili ve rengin belirlenebilir.

4.3.1 Border-Style

border-style uygulanacak elemente nasıl bir çerçeve uygulanacağını belirleyen özelliktir. Border-style özelliği aşağıdaki değerleri alabilir.

dotted	Noktalı bir kenarlık tanımlar
dashed	Kesikli bir kenarlık tanımlar
solid	Düz bir kenarlık tanımlar
double	Çift kenarlık tanımlar
groove	3 boyutlu yivli bir kenarlık tanımlar
ridge	3 boyutlu çıkıntılı bir kenarlık tanımlar
inset	3 boyutlu bir iç kenarlık tanımlar
outset	3 boyutlu bir dış kenarlık tanımlar
none	Kenarlık tanımlamaz
hidden	Gizli bir kenarlık tanımlar

Örnek

```
<!DOCTYPE html>
<html>
<head>
<style>
p.noktali {border-style: dotted;}
p.kesikli {border-style: dashed;}
p.duz {border-style: solid;}
p.cift {border-style: double;}
p.yivli {border-style: groove;}
p.cikintili {border-style: ridge;}
p.ic {border-style: inset;}
p.dis {border-style: outset;}
p.kenarliksiz {border-style: none;}
p.gizli {border-style: hidden;}
p.karisik {border-style: dotted dashed solid double;}
</style>
</head>
<body>
<p class="noktali">Noktalı kenarlık</p>
<p class="kesikli">Çizgili kenarlık</p>
<p class="duz">Düz kenarlık.</p>
<p class="cift">Çift kenarlık</p>
<p class="yivli">3 boyutlu yivli bir kenarlık </p>
<p class="cikintili">3 boyutlu çıkıntılı bir kenarlık</p>
<p class="ic">3 boyutlu bir iç kenarlık</p>
<p class="dis">3 boyutlu bir dış kenarlık</p>
<p class="kenarliksiz">Kenalıksız</p>
<p class="gizli">Gizli kenarlık</p>
<p class="karisik">Karşık kenarlık</p>
</body>
</html>
```

Noktalı kenarlık

Çizgili kenarlık

Düz kenarlık.

Çift kenarlık

3 boyutlu yivli bir kenarlık

3 boyutlu çıkıntılı bir kenarlık

3 boyutlu bir iç kenarlık

3 boyutlu bir dış kenarlık

Kenalıksız

Gizli kenarlık

Karşık kenarlık

4.3.2 Kenar Kalınlığı

Bir html elemanının dört kenarının kenarlık kalınlığını ayarlamak için kullanılan bir özelliktir. Değer olarak px, pt, cm, em türünden bir sayı alabileceği gibi thin, medium, thick şekilden üç değer de alabilir.

Örnek

```
p.one {  
  border-style: solid;  
  border-width: 5px;  
}  
  
p.two {  
  border-style: solid;  
  border-width: medium;  
}  
  
p.three {  
  border-style: dotted;  
  border-width: 2px;  
}  
  
p.four {  
  border-style: dotted;  
  border-width: thick;  
}
```

Birinci örnek

İkinci örnek

Üçüncü örnek

Dördüncü örnek

Not: "Border-width" özelliği tek başına kullanıldığında çalışmaz. Önce kenarlıkları ayarlamak için her zaman "border-style" özelliğini belirtmeniz gerekir.

Örnek

```
p.one {  
  border-style: solid;  
  border-width: 5px 20px; /* 5px üst ve alt, 20px yanlar */  
}  
p.two {  
  border-style: solid;  
  border-width: 20px 5px; /* 20px üst ve alt, 5px yanlar */  
}  
p.three {  
  border-style: solid;  
  border-width: 25px 10px 4px 35px; /* 25px üst 10px sağ 4px alt 35px sol */  
}
```

Birinci örnek

İkinci örnek

Üçüncü örnek

4.3.3 Kenarlık Rengi

Kenarlık rengini ayarlamak için border-color özelliği kullanılır. Renk değerleri için hexadecimal (#FF0000), rgb (rgb(255,0,0)), rgba (rgba(255,0,0,0.5)) veya isimli renkler (red, blue gibi) kullanılabilir.

Örnek

```
p.one {  
  border-style: solid;  
  border-width: 5px;  
  border-color: brown;  
}  
  
p.two {  
  border-style: solid;  
  border-width: 5px;  
  border-color: red green blue yellow;  
}  
  
p.three {  
  border-style: dotted;  
  border-width: 5px;  
  border-color: rgb(0, 200, 0);  
}
```

Kırmızı renkli

üst kırmızı
sağ yeşil
alt mavi
sol sarı

Not: "border-color" özelliği tek başına kullanıldığında çalışmaz. Önce kenarlıkları ayarlamak için her zaman "border-style" özelliğini belirtmeniz gerekir.

4.3.4 Kısa Kullanım (border)

CSS'de border özelliği, bir elemanın tüm kenarlarına çerçeve eklemek için kullanılan kısayol bir özelliktir. Bu özellik, çerçevenin genişliği (border-width), stili (border-style) ve rengi (border-color) olmak üzere üç değeri tek bir ifadede birleştirir. Bu kısayol, stil dosyanızı daha sade ve yönetilebilir hale getirir.

Sözdizimi => border: [border-width] [border-style] [border-color];

- border-width: Çerçevenin kalınlığını belirtir. Değerler piksel (px), em, rem gibi CSS ölçü birimleriyle verilebilir. Örneğin, 2px.
- border-style: Çerçevenin stilini tanımlar. Bu stil solid, dotted, dashed, double, groove, ridge, inset, ve outset gibi değerler alabilir.
- border-color: Çerçevenin rengini belirtir. Bu renk, renk isimleri (örneğin, red), hexadecimal değerler (örneğin, #ff0000), RGB (örneğin, rgb(255, 0, 0)) ya da RGBA (örneğin, rgba(255, 0, 0, 0.5)) gibi formatlarda olabilir.

Örnek

```
p.one {  
  border: 5px solid red;  
}  
p.two{  
  border-left: 5px solid red;  
}
```

Örnek 1

Örnek 2

4.3.5 Border-radius

border-radius özelliği HTML elemanlarının köşelerini yuvarlak yapmak için kullanılır. Bu özellik düğmeler, fotoğraflar, kutular ve diğer birçok element için daha yumuşak ve estetik bir görünüm sağlar. border-radius ile köşelerin yarıçapını belirleyerek, köşelerin ne kadar yuvarlak olacağını kontrol edebilirsiniz.

Sözdizimi => border-radius: [value];

border-radius özelliği bir veya birden fazla değer alabilir:

- Tek Değer: Tüm köşelere aynı yarıçap uygulanır.
- İki Değer: İlk değer üst sol ve alt sağ köşelere, ikinci değer üst sağ ve alt sol köşelere uygulanır.
- Üç Değer: İlk değer üst sol köşeye, ikinci değer üst sağ ve alt sol köşelere, üçüncü değer alt sağ köşeye uygulanır.
- Dört Değer: Sırasıyla üst sol, üst sağ, alt sağ ve alt sol köşelere uygulanır.

Değerler piksel (px), em, % gibi CSS boyut birimleriyle belirtilebilir. Yüzde değerleri kullanıldığında, yarıçap elementin genişliğine ve yüksekliğine göre orantılı olarak ayarlanır.

Örnek

```
p.normal {  
  border: 3px solid red;  
}  
p.round1 {  
  border: 3px solid red;  
  border-radius: 10px;  
}  
p.round2 {  
  border: 3px solid red;  
  border-radius: 5px 10px 15px 20px;  
}  
p.round3 {  
  border: 3px solid red;  
  border-radius: 50%;  
}
```

Normal Kenarlık

Yuvarlak Kenarlık

Farklı değerlerde kenarlar

Yüzde değerli kenarlık

4.4 Margin (Dış kenar boşluğu)

Margin özelliği bir HTML elemanının etrafındaki boşluğu (dış kenar boşluğu) belirlemek için kullanılır. Elemanların birbirlerinden ve sayfanın kenarlarından ne kadar uzakta yer alacağını belirlemek için kullanılır. margin özelliği sayfa düzeninin ayarlanmasında ve elemanların görsel olarak birbirinden ayrılmasında önemli bir rol oynar.

CSS'de dört tür margin özelliği bulunur:

- margin-top: Elemanın üstündeki dış kenar boşluğunu belirler.
- margin-right: Elemanın sağ tarafındaki dış kenar boşluğunu belirler.
- margin-bottom: Elemanın altındaki dış kenar boşluğunu belirler.
- margin-left: Elemanın sol tarafındaki dış kenar boşluğunu belirler.

Kısa tanımlama

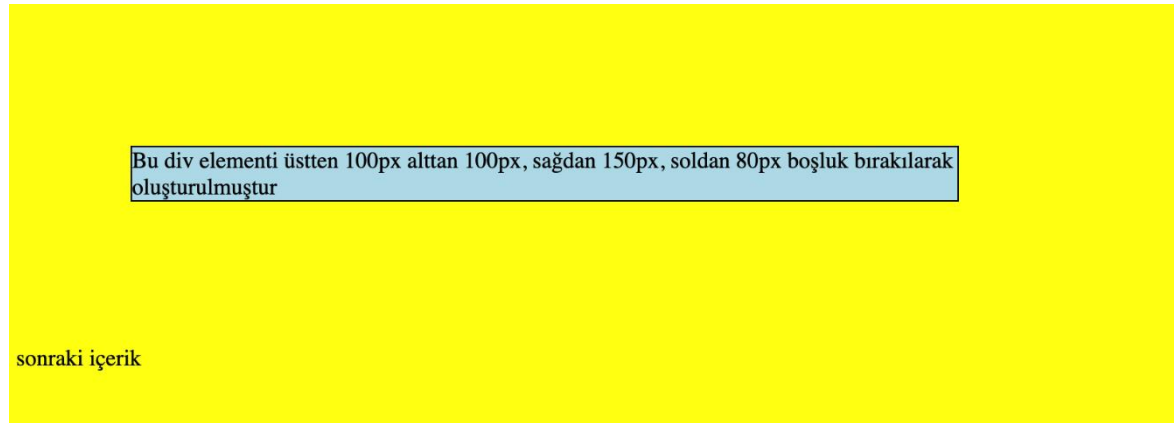
Ayrıca margin kısayol özelliği tüm kenar boşluklarını tek bir tanımla ayarlamanıza olanak tanır. Bu özellik bir değerden dört farklı değere kadar kullanılabilir:

- Tek Değer: margin: 10px; tüm kenarlarına 10px margin ekler.
- İki Değer: margin: 10px 20px; üst ve alt kenarlarına 10px, sağ ve sol kenarlarına 20px margin ekler.
- Üç Değer: margin: 10px 20px 30px; üst kenara 10px, sağ ve sol kenarlarına 20px, alt kenara 30px margin ekler.
- Dört Değer: margin: 10px 20px 30px 40px; sırasıyla üst, sağ, alt ve sol kenarlarına farklı margin değerleri ekler.

Örnek

```
<!DOCTYPE html>
<html>
<head>
<style>
body{
background-color: yellow;
}
div.bosluk {
border: 1px solid black;
margin-top: 100px;
margin-bottom: 100px;
margin-right: 150px;
margin-left: 80px;
background-color: lightblue;
}
</style>
</head>
<body>

<div class="bosluk">Bu div elementi üstten 100px alttan 100px, sağdan 150px,
soldan 80px boşluk bırakılarak oluşturulmuştur</div>
<p>sonraki içerik</p>
</body>
</html>
```



```
margin: 100px 150px 100px 80px;
```

Not: Bir önceki örneğin kısa kullanımı

Araştırın

1. Margin değeri negatif olabilir mi ?
2. Margin değerine auto girildiğinde nasıl bir sonuç alınır?

4.5 Padding

Padding özelliği bir HTML elemanın içeriği ile sınırı (border) arasındaki boşluğu belirlemek için kullanılır. Padding elemanın iç kısmındaki boşluğu artırarak, içerik ile sınır arasında görsel bir "alan" yaratır. Padding sayfa düzeninin ayarlanmasında ve içeriklerin daha okunabilir hale getirilmesinde önemli bir rol oynar.

padding ile ilgili dört temel özellik bulunmaktadır:

- padding-top: Elemanın üst kısmındaki iç boşluğu belirler.
- padding-right: Elemanın sağ tarafındaki iç boşluğu belirler.
- padding-bottom: Elemanın alt kısmındaki iç boşluğu belirler.
- padding-left: Elemanın sol tarafındaki iç boşluğu belirler.

Kısa tanımlama

padding özelliği aynı zamanda bir kısayol olarak da kullanılabilir. Bu sayede elemanın tüm kenarlarına tek bir komutla padding ekleyebilirsiniz. Bu kısayol özelliği bir değerden dört farklı değere kadar kullanılabilir:

- Tek Değer: padding: 10px; tüm kenarlara 10px padding ekler.
- İki Değer: padding: 10px 20px; üst ve alt kenarlara 10px, sağ ve sol kenarlara 20px padding ekler.
- Üç Değer: padding: 10px 20px 30px; üst kenara 10px, sağ ve sol kenarlara 20px, alt kenara 30px padding ekler.

- Dört Değer: padding: 10px 20px 30px 40px; sırasıyla üst, sağ, alt ve sol kenarlara farklı padding değerleri ekler.

Örnek

```
<!DOCTYPE html>
<html>
<head>
<style>
body{
background-color: yellow;
}
div.bosluk {
border: 1px solid black;
padding-top: 100px;
padding-bottom: 100px;
padding-right: 150px;
padding-left: 80px;
background-color: lightblue;
}
</style>
</head>
<body>

<div class="bosluk">Bu div elementi üstten 100px alttan 100px, sağdan 150px,
soldan 80px iç boşluk bırakılarak oluşturulmuştur</div>
<p>sonraki içerik</p>
</body>
</html>
```

Bu div elementi üstten 100px alttan 100px, sağdan 150px, soldan 80px boşluk bırakılarak oluşturulmuştur

sonraki içerik

padding: 100px 150px 100px 80px;

Not: Bir önceki örneğin kısa kullanımı

Araştırın

1. Padding değeri negatif olabilir mi ?

4.6 Genişlik ve Yükseklik

Width (genişlik) ve height (yükseklik) özellikleri bir HTML elemanının boyutlarını kontrol etmek için kullanılır. Bu özellikler elemanların sayfa içerisinde kapladıkları alanı belirlemek için oldukça önemlidir ve web tasarımının temel taşlarındandır.

Width (Genişlik)

width özelliği bir elemanın genişliğini belirler. Genellikle piksel (px), em, rem, % (yüzde) gibi CSS birimleri ile tanımlanır. Yüzde değeri (%) elemanın genişliğini onu içeren ebeveyn elemanın genişliğine göre bir oran olarak ayarlar. Bu özellik responsive (duyarlı) tasarım yaparken çok kullanışlıdır. Çünkü elemanların boyutları ekranın boyutuna göre değişebilir.

max-width ve min-width özellikleri elemanın genişliğinin belirli bir maksimum veya minimum değere ulaşabileceği sınırları belirler. Bu özellikler farklı ekran boyutları için tasarımların daha esnek olmasını sağlar.

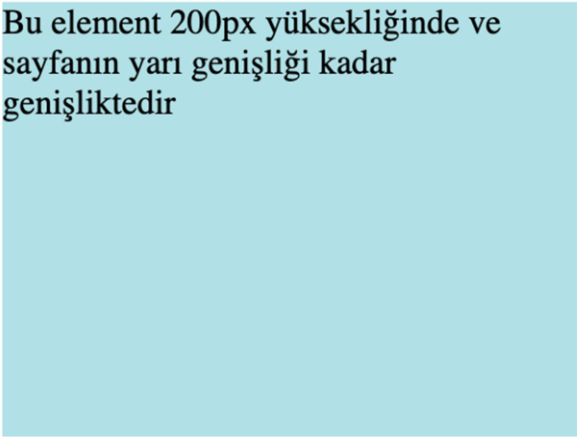
Height (Yükseklik)

height özelliği bir elemanın yüksekliğini belirler. Bu özellik de genellikle px, em, rem, % gibi birimlerle tanımlanır. % değeri elemanın yüksekliğini onu içeren ebeveyn elemanın yüksekliğine göre oransal olarak ayarlar. max-height ve min-height özellikleri elemanın yüksekliğinin belirli bir maksimum veya minimum değere ulaşabileceği sınırları belirler.

Örnek

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  height: 200px;
  width: 50%;
  background-color: powderblue;
}
</style>
</head>
<body>
<div>Bu element 200px yüksekliğinde ve sayfanın yarı genişliği kadar
genişliktedir</div>
</body>
</html>
```

Bu element 200px yüksekliğinde ve sayfanın yarı genişliği kadar genişliktedir



Örnek

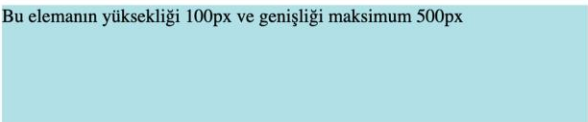
```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  max-width: 500px;
  height: 100px;
  background-color: powderblue;
}
</style>
</head>
<body>

<div>Bu elemanın yüksekliği 100px ve genişliği maksimum 500px</div>

<p>Tarayıcı penceresini boyutlandırarak gözlemleyebilirsiniz</p>

</body>
</html>
```

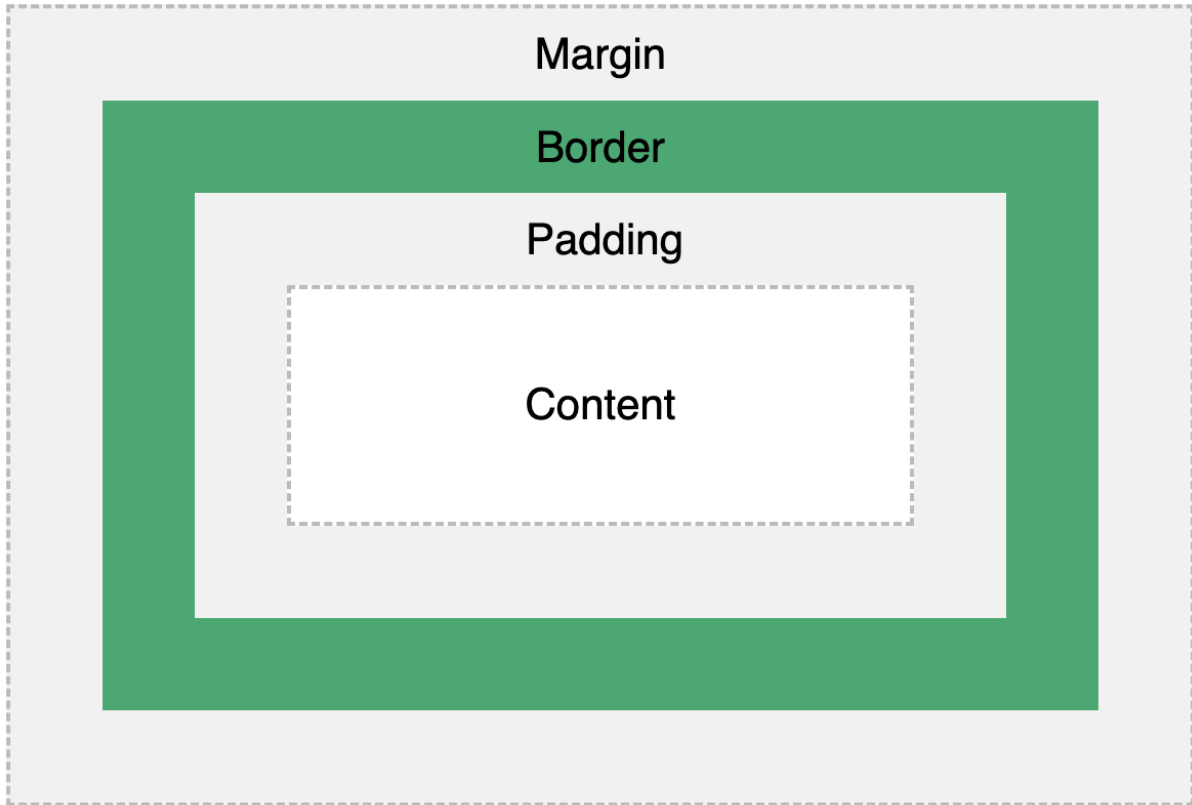
Bu elemanın yüksekliği 100px ve genişliği maksimum 500px



Tarayıcı penceresini boyutlandırarak gözlemleyebilirsiniz

4.7 Kutu Modeli

Kutu modeli web sayfalarının düzenini ve tasarımını oluştururken kullanılan temel bir kavramdır. Her HTML elemanı bir kutu olarak düşünülür ve bu model kutuların nasıl işlendiğini ve birbirleriyle nasıl etkileşime girdiğini tanımlar. Box model; elemanın içeriği, padding'i (iç boşluk), border'ı (sınır) ve margin'i (dış boşluk) olmak üzere dört ana bölümden oluşur.



Kutu modeli elemanların etrafına kenarlık eklememize ve elemanlar arasındaki boşluğu tanımlamamıza olanak tanır.

Örnek

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  background-color: lightgrey;
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
}
</style>
</head>
<body>

<h2>Kutu Modeli Örneği</h2>

<p>CSS kutu modeli aslında her HTML ögesinin etrafını saran bir kutudur.
Şunlardan oluşur: kenarlıklar, dolgular, kenar boşlukları ve gerçek
içerik.</p>

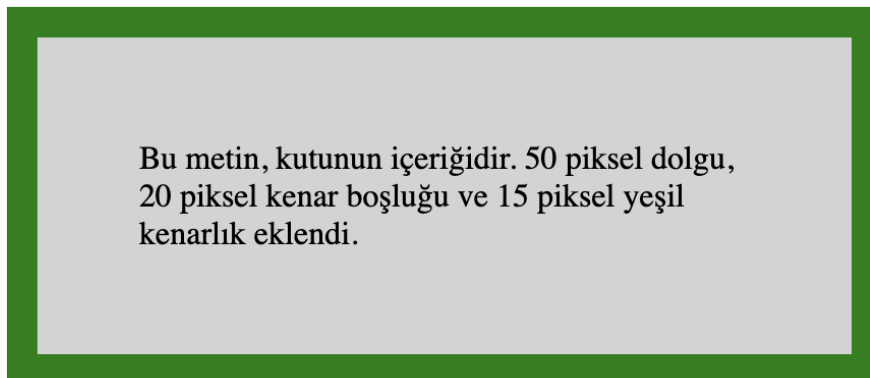
<div>Bu metin, kutunun içeriğidir. 50 piksel dolgu, 20 piksel kenar boşluğu
ve 15 piksel yeşil kenarlık eklendi.</div>

<p>CSS kutu modeli aslında her HTML ögesinin etrafını saran bir kutudur.
Şunlardan oluşur: kenarlıklar, dolgular, kenar boşlukları ve gerçek
içerik.</p>

</body>
</html>
```

Kutu Modeli Örneği

CSS kutu modeli aslında her HTML ögesinin etrafını saran bir kutudur. Şunlardan oluşur: kenarlıklar, dolgular, kenar boşlukları ve gerçek içerik.



CSS kutu modeli aslında her HTML ögesinin etrafını saran bir kutudur. Şunlardan oluşur: kenarlıklar, dolgular, kenar boşlukları ve gerçek içerik.

Önemli: Bir öğenin genişlik ve yükseklik özelliklerini CSS ile ayarladığınızda içerik alanının genişliğini ve yüksekliğini ayarlamanız yeterlidir. Bir öğenin toplam genişliğini ve yüksekliğini hesaplamak için dolgu ve kenarlıkları da dahil etmeniz gerekir.

Örnek

```
div {  
  width: 320px;  
  height: 50px;  
  padding: 10px;  
  border: 5px solid gray;  
  margin: 0;  
}
```

Bu <div> öğesinin toplam genişliği 350 piksel, toplam yüksekliği ise 80 piksel olacaktır:

320 piksel (içerik alanının genişliği)
+ 20 piksel (sol dolgu + sağ dolgu)
+ 10 piksel (sol kenar + sağ kenarlık)
= 350 piksel (toplam genişlik)

50 piksel (içerik alanının yüksekliği)
+ 20 piksel (üst dolgu + alt dolgu)
+ 10 piksel (üst kenarlık + alt kenarlık)
= 80 piksel (toplam yükseklik)

Bir elemanın toplam genişliği şu şekilde hesaplanmalıdır:

Toplam öğe genişliği = genişlik + sol dolgu + sağ dolgu + sol kenarlık + sağ kenarlık

Bir elemanın toplam yüksekliği şu şekilde hesaplanmalıdır:

Toplam eleman yüksekliği = yükseklik + üst dolgu + alt dolgu + üst kenar + alt kenar

Not: Dış kenar boşluğu (margin) özelliği aynı zamanda kutunun sayfada kaplayacağı toplam alanı da etkiler ancak kenar boşluğu kutunun gerçek boyutuna dahil edilmez. Kutunun toplam genişliği ve yüksekliği sınırdadır.

4.8 Outline

Outline özelliği HTML elemanlarının dışında ekstra bir çizgi (outline) çizmeye yarar. Bu çizgi elemanın boyutunu etkilemez ve CSS Box Model'inin dışında yer alır. outline genellikle kullanıcı arayüzünde odaklanma (focus) durumunu belirtmek için kullanılır ancak görsel stilizasyon amacıyla da kullanılabilir.

Online ile ilgili dört temele özellik vardır.

- **outline-color:** Outline'ın rengini belirler. Standart renk değerleri kullanılabilir (örneğin, red, #FF0000, rgba(255, 0, 0, 0.5), vb.). Özel bir durum olarak, invert değeri mevcuttur. Bu değer outline renginin arka plana göre otomatik olarak ayarlanmasını sağlar. Ancak destek ve kullanımı tarayıcıya bağlıdır.
- **outline-style:** Outline'ın stilini belirler. solid, dotted, dashed, double, groove, ridge, inset, outset gibi değerleri alabilir. Eğer bir elemana outline uygulamak istiyorsanız bu özellik mutlaka tanımlanmalıdır çünkü varsayılan değeri none'dır.
- **outline-width:** Outline'ın kalınlığını belirler. Değerler thin, medium, thick veya piksel cinsinden (örneğin, 2px) olabilir.
- **outline-offset:** Outline'ın elemandan ne kadar uzakta çizileceğini belirler. Pozitif değerler outline'ı elemandan dışarı doğru ittirirken negatif değerler elemanın içine doğru çeker.

Özellik outline-style, anahattın stilini belirtir ve aşağıdaki değerlerden birine sahip olabilir:

- dotted- Noktalı bir taslak tanımlar
- dashed- Kesikli bir taslak tanımlar
- solid- Sağlam bir taslak tanımlar
- double- Çift taslak tanımlar
- groove- 3 boyutlu yivli bir taslağı tanımlar
- ridge- 3 boyutlu çıkıntılı bir taslağı tanımlar
- inset- 3 boyutlu bir iç metin taslağını tanımlar
- outset- 3 boyutlu başlangıç taslağını tanımlar
- none- Hiçbir taslak tanımlamaz
- hidden- Gizli bir taslak tanımlar

Örnek

```
<!DOCTYPE html>
<html>
<head>
<style>
p.ex1 {
  border: 10px solid blue;
  outline-style: solid;
  outline-color: red;
  outline-width: 4px;
}
</style>
</head>
<body>

<p class="ex1">Online örneği</p>

</body>
</html>
```

Outline örneği

Kısa kullanım

Örnek

```
p.ex1 {outline: dashed;}
p.ex2 {outline: dotted red;}
p.ex3 {outline: 5px solid yellow;}
p.ex4 {outline: thick ridge pink;}
```

outline-offset bir öğenin outline ile border arasına boşluk ekler. Öğe ile onun outline'ı arasındaki boşluk şeffaftır.

Aşağıdaki örnek paragrafın kenarının 15 piksel dışına bir outline çizer.

Örnek

```
<!DOCTYPE html>
<html>
<head>
<style>
body{
  background:#eeeeee;
}
p {
  margin: 30px;
  background:yellow;
  border: 1px solid black;
  outline: 1px solid red;
  outline-offset: 15px;
}
</style>
</head>
<body>
<p>bu paragrafın kenarının 15px dışında bir outline vardır.</p>
</body>
</html>
```

bu paragrafın kenarının 15px dışında bir outline vardır.

