CSS NEDİR?

- CSS **"Cascading Style Sheets"** kelimesinin baş harflerinden oluşur. Dilimizdeki anlamı şudur: **Basamaklı Stil Sayfası.**
- CSS, bir HTML elementinin nasıl görüneceğini belirler. Siteyi görselleştirir. Örneğin web sitemizdeki bir yazının rengini, bir görselin şeklini, arkaplan rengini ve birçok özelliği değiştirmek istediğimizde css koduyla oynayarak bir hamlede işimizi halledebiliriz.
- CSS, HTML dosyamızın daha anlaşılır olmasını sağlar. CSS kodları tek bir dosyada toplanır ve kolaylıkla okunur ve değiştirilir.
- CSS dosyalarımızın uzantısı **.css** dir. CSS dosyası birçok stil barındırabilir. Yani bir yazının rengiyle aynı zamanda büyüklüğünü hatta yazı stilini aynı anda belirleyebiliriz.
- HTML ilk ortaya çıktığında siyah beyaz bir televizyondan farksızdı. Her zaman aynı kodlar kullanılır ve aynı biçimde yazılar görünürdü. Örneğin bir h1 etiketiyle başlık yazıp farklı bir yerdeki h1 elementini değişik bir renk ve tarzda gösteremiyorduk. CSS kodlarıyla bu sorundan kurtulmuş olduk.
- CSS'i etkin kullanabilmek için her html öğesinin etrafında görünmeyen bir kutu olduğunu hayal etmek işe yarayabilir.

CSS KOD YAPISI:

- CSS kodları şu örnekteki gibi yazılır. ***p { color:blue; }***
- İlk önce değişiklik yapmak istediğimiz etiketi yazdık. Yukarıdaki örnekte **p** ifadesiyle paragraflar için bir değişiklik yapacağımızı belirtiyoruz.
- Daha sonra **süslü parantezler** içinde hangi özelliği kullanacağımızı belirliyoruz.
- Yukarıdaki örnekteki **color** ifadesiyle tüm paragrafların rengini değiştirmek istediğimizi belirtiyoruz.
- İki noktadan sonra ise yapmak istediğimiz değişikliğin özelliğini belirtiyoruz. Yukarıdaki örnekte **blue** ifadesiyle paragrafların mavi renkte olmasını istedik.
- Son olarak **noktalı virgülle** css kodumuzu tamamladık.
- Ancak yukarıda sadece bir özellik belirttik. Bunu çoğaltabiliriz.
- **Örnek:**

```
p {  color:blue;
     font-size:12px;
     text-align:center;
  }
```

- Yukarıdaki örnekte paragraflara renk özelliğinin yanı sıra **font-size** özelliği ile font büyüklüğünü ve **text-align** özelliği ile de metin hizalama ayarlarını yaptık. (**center;** metinleri ortalar)

CSS'TE AÇIKLAMA KULLANMAK:

- Her programlama dilinde olduğu gibi ve HTML dilinde de bazen kullanıcıların görmeyeceği ancak bizim için referans olacak açıklamalar yazmak gerekebilir. Bunun için açılış etiketi olarak **/*** karakterleri kapanış etiketi olarak da ***/** karakterleri kullanılmaktadır:

- **Örnek:**

```
p { color:blue;          /* Renk mavi olacak*/
    font-size:12px;      /* Font büyüklüğü 12px olacak*/
    text-align:center;   /* Metin ortalanmış olacak*/
  }
```

CSS'TE YAZIM TÜRLERİ:

- Css kodlarımızı farklı amaçlara göre 3 farklı şekilde tanımlayabiliriz:
  1. **Inline Css**
  2. **Internal Css**
  3. **External Css**

- **Inline Css:**
  - Sadece tanımlandığı etiket için geçerli olmasını istediğimiz css kodlarımızı inline css olarak tanımlarız.
  - Inline css kodları direkt etiket içine yazılır ve sadece o etikete uygulanır.
  - Inline olarak yazılan css kodları aynı özellik için yazılmış diğer css kodunu ezer ve etkin olur. Yani daha sonra göreceğimiz internal css ve external css kodları geçersiz olur.
  - Örnek inline css yazılışı: ***<h1 style="color:red;">*** *Merhaba* ***</h1>***

- **Internal Css:**
  - Internal css **<head> </head>** etiketleri arasına eklenerek kullanılır.
  - Internal olarak tanımlanan css kodları sadece bulunduğu html sayfasını ilgilendirir.
  - Örnek internal css yazılışı:

```
<style type="text/css">
        h1 {
           font-size :32px;
           color :blue;

        }
</style>
```

- **External Css:**
  - Eğer tanımladığımız css kodları birden fazla html sayfasını ilgilendiriyor ise bu durumda **.css** uzantılı bir dosya oluşturup bu dosya içerisine css kodlarımızı tanımlamamız gerekir.
  - Harici .css uzantılı dosyaya yazdığımız css kodlarını hangi html sayfasına eklemek istiyorsak o html sayfası ile css dosyasını birbirine **<link>** ile bağlamamız gerekiyor. Bunun için **<head> </head>** etiketleri arasına aşağıdaki gibi bir ifade yazmalıyız. Örnek:
    <link rel="stylesheet" href="style.css">
  - Yukarıdaki örnekte href etiketinin içine harici olarak oluşturduğumuz css dosyasının adını yazmalıyız.

METİN İŞLEMLERİ:

- ***font-size : 16px;*** (**Yazı boyutu**)
  - Varsayılan **font-size değeri 16px 'dır.**
  - Varsayılan font-size değerini body etiketine css özelliği vererek değiştirebiliriz:

- Örnek: ***body { font-size: 20px }***

- **px** yerine **em** kullanırsak varsayılan font büyüklüğe oranla bir punto belirtmiş oluruz. Örneğin varsayılan font-size değeri 16px olsun, biz bu durumda herhangi bir etikete 2em değerini verirsek font büyüklüğünün iki katı yani 32px olarak belirtmiş oluruz.
- **font-size : 2em; = font-size : 32px;**

- ***font-family : "Times New Roman", Times, serif;*** (**Font türü) ;**
  - Burada "Times New Roman" font türü uygulanır ancak bu font türünü tarayıcı desteklemiyorsa bu durumda ikinci font türü olan **Times** ya da bir sonraki **serif** font türü etkin olacaktır.
  - "Times New Roman" font türünün tırnaklar içinde belirtilmesindeki sebep birden fazla kelimeden oluşmasıdır.
  - Bu arada varsayılan font-family değeri **"Times New Roman"** 'dır.
  - Değişik tarzda font-family türleri için **https://fonts.google.com/** sitesindeki örnekleri kullanabiliriz.
- ***font-weight : bold;*** (**Yazı kalınlığı)** ; Alabileceği değerler: **normal, bold, bolder**
- ***text-decoration: underline;*** (**Metinlerin altını, üstünü vbs. çizip çizmeyeceğimizi belirler)**
  - **underline**; altını çizer
  - **overline**; üstünü çizer
  - **line-through**; ortasını çizer
  - **none:** bütün özellikleri kaldırır
- ***text-transform: uppercase;*** (**Metinlerin büyük küçük harf ayarlarını yapar)**
  - **uppercase**; büyük harfe çevirir
  - **lowercase**; küçük harfe çevirir
  - **capitalize**; her kelimenin baş harfini büyük harfe çevirir
- ***text-indent: 5px;*** (**Paragrafın ilk cümlesinin ne kadar içeriden başlayacağını belirtir)**
- ***text-align: center;*** (**Yazıların hizalanmasını sağlar.)**
  - **left | center | right | justify (iki yana yasla) gibi değerler alır.**
- ***letter-spacing: 1.2px;*** (**Harfler arasındaki mesafeyi belirtir.**
- ***line-height: 0.7;*** (**Satır yüksekliğini belirtir)**
- ***word-spacing: 5px;*** (**Kelimeler arasındaki boşluğu belirtir; Negatif bir değer alırsa kelimeler birbirine çok yaklaşır.**
- ***color : red;*** (**Yazı rengi)** ;
  - Yazı rengi için önceden tanımlanmış renk isimlerini kullanabiliriz. Burada yazımızı kırmızı yaptık.
  - Ayrıca rengimizi RGB türünde belirtebiliriz. **rgb (Red,green,blue)** Üç ana temel renk için **0-255** arasında değer alır. Örnek: **color: rgb (190,94,244);**
  - Bir başka renk belirleme yöntemi ise **Hexedecimal** renk tanımlamasıdır. Örnek: **color: #ffffff;**

- **Not: Sitemizde ikonlar kullanmak istiyorsak https://fontawesome.com/ gibi internet sitelerinden faydalanabiliriz.**

- **Not: Paragraflarımızı sütunlara bölmek istiyorsak column-count özelliğini kullanabiliriz.**
  - ***column-count: 2*** **(paragraf iki sütuna böler)**
  - ***column-rule: solid 1px gray*** **(sütun kenarlıklarını düz çizgi, 1px ve gri renkte yapar)**
  - ***column-gap: 2em*** **(sütunlar arası uzaklığı belirler)**
  - ***column-span: all***
    column-count ile oluşturduğumuz sütunlarda bulunan bir etiketin örneğin bir başlığın tüm sütunları kaplamasını isteyebiliriz. Bu durumda *column-span: all;* değeri vererek başlık etiketlerinin ayrı bir şekilde konumlandırılmasını sağlayabiliriz. Başlıktan sonra gelen metin yine aynı sütun sayısı üzerinden bölünmeye devam eder.
  - ***column-fill:*** ***Sütunların eşitliğini ayarlar. İki değer alır; balance veya auto;***
    Ör1: column-fill: balance;
    her kolonu eşit yükseklikte olacak şekilde ayarlar. Zaten varsayılan başlangıç değeridir.
    Ör2: column-fill: auto;
    sütunları sıralı olarak doldurur. Bu özelliğin çalışması için html elemanına yükseklik değeri verilmesi gerekir. Belirlenen yükseklik değerine göre kolonları doldurur ve son kolona geri kalan içerik yerleştirilir, bazen kolonlar boşta kalabilir.
  - ***column-width:*** ***kolonların genişliğini belirler.*** **Ör:**
    Ör: *column-width: 250px*
    bu durumda sayfamız 250'şer piksellik sütunlara bölünecektir.)
  - ***columns:*** ***responsive özelliği vermek için kullanılır.*** **Ör:**
    Ör: *columns: 300px 3*
    bu durumda sayfamız 3 sütuna ayrılacak ama sayfa küçüldüğünde 3 sütunluk özelliğini kaybedecek ve minimum 300px yazdığımız için sütun sayısı gittikçe küçülecek ve en son bir sütun kalacaktır.
  - **Not: column özelliği css3 ile geldiği için tüm tarayıcılara uygun hale gelmesi için şu işlemleri yapmalıyız: Ör:**
    - ***-webkit-column-count: 2*** **(chrome-safari için)**
    - ***-moz-column-count: 2*** **(mozilla firefox için)**
    - ***-o-column-count: 2*** **(opera için)**
    - ***-ms-column-count: 2*** **(internet Explorer için)**

LİSTE İŞLEMLERİ:

list-style-type : liste işaretini belirler.

- ***list-style-type: none ;*** → **liste imleri gözükmeyecek.**
- ***list-style-type: disc;*** → **disk işareti koyar**
- ***list-style-type: circle;*** → **çember şeklinde işaret koyar**
- ***list-style-type: square;*** → **kare şeklinde işaret koyar**
- ***list-style-type: decimal;*** → **sıralı rakam koyar**
- ***list-style-type: lower-roman;*** → **sıralı küçük roma rakamı koyar (i, ii, iii ,iv …)**
- ***list-style-type: upper-roman;*** → **sıralı büyük roma rakamı koyar (I, II, III, IV …)**
- ***list-style-type: lower-alpha;*** → **sıralı küçük harf koyar (a, b, c, d…)**

- ***list-style-type: upper-alpha;*** → **sıralı büyük harf koyar (A, B, C, D…)**

list-style-image: liste işareti yerine bir image koyar

- ***list-style-image: url('kiraz.jpg');***

list-style-position: liste işareti içeriden mi dışarıdan mı başlayacak onu belirler.

- ***list-style-position: inside*** **; metnin içerisinden başlayacak**
- ***list-style-position: outside*** **metnin dışından başlayacak**

NESNELERİN YÜKSELİK VE GENİŞLİK AYARLARI:

- ***width: 600px;*** **(Elemanların genişlik değeri)**
- ***height: 600px;*** **(Elemanların yükseklik değeri)**
- ***max-width: 600px;*** **(Bir nesnenin alabileceği maksimum genişlik değeri için kullanılır)**
- ***max-height: 600px;*** **(Bir nesnenin alabileceği maksimum yükseklik değeri için kullanılır)**

KENARLIK İŞLEMLERİ:

- ***border: solid 1px red;*** **(Bir nesneye kenarlık eklemek için kullanılır)**
  - Border' a verebileceğimiz üç özellik vardır.
  - Yukarıdaki örnekte **solid** özelliği **kenarlık stilini** belirtir ve bu özelliği yazmamız zorunludur aksi taktirde kenarlık eklenmez. Solid düz çizgi kenarlığı eklememizi sağlar ancak solid dışında başka tercihler de vardır. Bunlar şu şekildedir:
    - **dotted:** noktalı kenar
    - **dashed:** kesik çizgili kenar
    - **double:** çift çizgili kenar
    - **groove:** Üç boyutlu kenar
    - **ridge:** Üç boyutlu düz kenar
    - **inset:** Üç boyutlu iç kenar
    - **outset:** Üç boyutlu dış kenar
    - **hidden:** kenarlık vardır ama görünmez
    - **none:** kenarlık eklemez
  - Yukarıdaki örnekte **1px** özelliği **kenarlık genişliğini** belirtir ancak bu özelliği yazmak zorunlu değildir yine de bu özelliği kullanmamız kendi tercihimizi kullanmak açısından iyi olur.
  - Yukarıdaki örnekte **red** özelliği **kenarlık rengini** belirtir ancak bu özelliği yazmak zorunlu değildir yine de bu özelliği kullanmamız kendi tercihimizi kullanmak açısından iyi olur.
  - Bunların dışında yukarıdaki özellikleri ayrı ayrı da belirtebiliriz. Bunun için aşağıdaki özellikleri belirtmemiz gerekir. Burada da **kenar stilini** belirtmemiz zorunludur.
    Örnekler:
    - ***border-style: solid;*** **kenarlık stili için**
    - ***border-width : 5px;*** **kenarlık genişliği için**

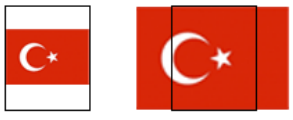
- ***border-color: green;*** **kenarlık rengi için**

- Bunların dışında kenarlığın sadece bir tarafına özellik verebiliriz**. Örnekler:**
  - ***border-top: solid 2px blue;***
    - Yukarıdaki css koduyla kenarlığın sadece **üst** kısmına özellik vermiş olduk. Burada da yine **kenar stilini** belirtmemiz zorunludur.
  - ***border-right: solid 2px blue;***
    - Yukarıdaki css koduyla kenarlığın sadece **sağ** kısmına özellik vermiş olduk. Burada da yine **kenar stilini** belirtmemiz zorunludur.
  - ***border-bottom: solid 2px blue;***
    - Yukarıdaki css koduyla kenarlığın sadece **alt** kısmına özellik vermiş olduk. Burada da yine **kenar stilini** belirtmemiz zorunludur.
  - ***border-left: solid 2px blue;***
    - Yukarıdaki css koduyla kenarlığın sadece **sol** kısmına özellik vermiş olduk. Burada da yine **kenar stilini** belirtmemiz zorunludur.
- Ayrıca kenarlıkların her birini spesifik olarak kullanabiliriz. **Örnekler:**
  - ***border-top-style:*** **üst kenarlığın stili**
  - ***border-top-width:*** **üst kenarlığın kalınlığı**
  - ***border-top-color:*** **üst kenarlığın rengi**
  - ***border-bottom-style:*** **alt kenarlığın stili**
  - ***border-bottom-width:*** **alt kenarlığın genişliği**
  - ***border-bottom-color:*** **alt kenarlığın rengi**
  - ***border-left-style:*** **sol kenarlığın stili**
  - ***border-left-width:*** **sol kenarlığın genişliği**
  - ***border-left-color:*** **sol kenarlığın rengi**
  - ***border-right-style:*** **sol kenarlığın stili**
  - ***border-right-width:*** **sol kenarlığın genişliği**
  - ***border-right-color:*** **sol kenarlığın rengi**
- Kenarların köşelerini yuvarlamak istiyorsak **border-radius** özelliğini kullanmamız gerekir. Dört ayrı köşe için ayrı ayrı yuvarlama özelliği verebiliriz. İlk değer **üst-sol**, ikinci değer **üst-sağ**, üçüncü değer **alt-sağ** ve dördüncü değer ise **alt-sol** köşeyi temsil eder. Örnek:
  - ***border-radius: 2px 4px 3px 6px***
    - Yukarıdaki örnekte **üst-sol köşe 2px, üst sağ köşe 4px, alt-sağ köşe 3px, alt-sol köşe ise 6px değerinde yuvarlatılacaktır.**
  - ***border-radius: 2px***
    - Tek bir değer varsa **bütün kenarlar** aynı oranda yuvarlatılır.
  - ***border-radius: 2px 4px***
    - İki değer varsa **ilk değer üst-sol ve alt-sağ, ikinci değer ise üst-sağ ve alt-sol** kenarı temsil eder.

ARKA PLAN İŞLEMLERİ:

- Arkaplan işlemleri genel olarak **body** yani sitemizin gövdesine veya **div, tablo** gibi nesnelere uygulanır. **Örnekler;**
- ***background-color: red;*** **(Arka plan rengi)**

- ***background-image: url ('resim.jpg');*** **(Arka plan resmi)**
- ***background-repeat:*** Kullandığımız arkaplan resminin tekrar edip etmeyeceğini belirtmemizi sağlar. Dört kullanımı vardır:
  - ***background-repeat: no-repeat;*** **resim tekrar edilmeyecek (döşenmeyecek)**
  - ***background-repeat: repeat;*** **resim her yere döşenecek**
  - ***background-repeat: repeat-x;*** **resim sağa doğru döşenecek**
  - ***background-repeat: repeat-y;*** **resim aşağıya doğru döşenecek**
- ***background-attachment:*** Kullandığımız arkaplanın sabit kalıp kalmayacağı hakkında ayar yapmamızı sağlar. Eğer sabit kalmasını istiyorsak **fixed** özelliği kullanırız. **Örnek:**
  - **background-attachment: fixed;**
- ***background-position:*** Bu özellik yardımıyla arkaplan resminin hizalanma şeklini belirleme şansına sahip oluruz. **Örnek:**
  - ***background-position:*** **right top;** **nesne sağ üst köşede yer alacaktır.**
- Yukarıdaki tüm özellikleri aynı anda kullanma şansımız da vardır. **Örnek;**
  - ***background:*** **blue url ('resim.jpg') no-repeat right top;**
- ***background-size:*** Arkaplan resmine genişlik yükseklik ayarı verebiliriz. **Örnekler:**
  - ***background-size: 300px;*** **300px hem genişlik hem yükseklik verdik.**
  - ***background-size: 300px 400px;*** **300px genişlik, 400px yükseklik verdik.**
  - ***background-size: cover***
    - **cover** arkaplan resminin genişlik veya yükseklik değerinden en küçük olanına göre arkaplanı kaplar ve böylece resim tüm arkaplanı kaplayacak şekilde ayarlanır ancak resmin kalan kısmı kesik görünür.
  - ***background-size: contain;***
    - **contain** arkaplan resminin genişlik veya yüksekliğinden en büyük olanına göre arkaplan alanın içine uydurur. Resmi herhangi bir yerini kesmeden tamamını gösterir, ancak bazı bölgeler arkaplan resmi olmadan görünür.



contain cover

ID VE CLASS İŞLEMLERİ:

id işlemleri:

- Etiketlere verdiğimiz benzersiz isme **id** denir. Sadece bir etikete özellik vermek istediğimizde kullanılır. Bir html dokümanında aynı id ismiyle sadece bir etiket olmalıdır.
  id ile seçim yaparken **#** işareti kullanılır. **Örnek:**
- id adını ilk başta html etiketini yazıyoruz**.** İstediğimiz adı kullanabiliriz.
  - ***<h1 id="baslik"> sayfa başlığı </h1>***
- Daha sonra css kodunun başına # işareti koyuyoruz.
  - ***#baslik { color: blue; }***
- Bundan sonra artık **baslik** adında başka id ismi kullanamayız.

class işlemleri:

- İstediğimiz elemanlara toplu özellikler verebilmek için sınıf oluşturmamız gerekir. İstediğimiz kadar class ismi kullanabiliriz. Bir etikete birden fazla class ismi verebiliriz. class ile seçim yaparken **.(nokta) işareti** kullanılır. **Örnek:**
  - ***.blue { background-color: blue }*** **blue adında bir class oluşturduk**
- Artık bu class'ı istediğimiz etiketlere uygulayabiliriz. Örnek:
  - *** MERHABA *** arkaplan rengi mavi olan bir div oluşturduk.
- Şimdi de başka bir class oluşturalım.

  .yuvarlak { border-radius: 50px } yuvarlak adında bir class oluşturduk.

- Şimdi önceki class'ı ve bu class'ı aynı yerde kullanalım.
  - *** MERHABA ***
- Böylece arkaplan rengi mavi olan ve köşeleri yuvarlatılmış bir div elde etmiş olduk.

CSS SEÇİCİLERİ:

Örnek:	**Açıklama:**
*	İstisnasız tüm etiketleri kapsar.
h1 , p	Tüm h1 ve p etiketlerini kapsar.
.deneme	Sınıf değeri olan tüm etiketleri kapsar.
#kutu	id değeri kutu olan tüm etiketleri kapsar.
p.deneme	Sadece sınıf özelliği deneme olan tüm paragrafları kapsar.
div p	div etiketi içinde olan tüm p etiketlerini kapsar.
div>p	Üst etiketi div olan tüm p etiketlerini kapsar.
div~p	div etiketinden sonra gelen aynı seviyedeki p etiketleri
div+p	div etiketinden sonra gelen aynı seviyedeki ilk p etiketleri

CSS SIRALAMA ÖNCELİĞİ

- Eğer iki veya daha fazla aynı özellikten verilmişse en sonuncusu geçerlidir.
  Ör: Aşağıda verilen örnekte aynı etikete farklı özellikler verilmiştir. Bunlardan en sonuncusu yani en alttaki geçerli olacağı için tüm h1 etiketlerinin rengi kırmızı olur.

```
<style type="text/css">
    h1 {
        color :blue;
      }
    h1 {
        color :red;
      }
</style>
```

- Eğer bir özellik diğerinden daha özelse, özel olan genel olandan daha önceliklidir.
  Ör: Aşağıda verilen örnekte aslında aynı etiketler hedef alınmıştır. Ama birinde yıldız seçicisi ile genel bir seçim yapılırken p seçicisi ile sadece paragrafları kastetmiş oluyoruz. Bu durumda p seçicisin özellikleri geçerli olur. Yani yazı rengi sarı olacaktır.

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

- css yazma stillerinde öncelik sırası şu şekildedir. **inline-internal-external**

NESNELERİ HİZALAMA

- Nesneleri hizalama için **float** özelliği kullanılır. Genelde div'leri yan yana getirmek için kullanırız. Bunun yanında resim hizalamalarında da kullanırız.

  float değerleri;
  - ***float: left;*** **sola hizalar.**
  - ***float: right;*** **sağa hizalar.**
  - ***float: none;*** **hizalama yapmaz.**

- **overflow** özelliği**:** Bir içeriğin kapsadığı alandan taşması durumunda (örneğin bir div'in içinde bulunan bir resmin veya bir paragrafın div'in dışına taşması) ne yapılacağını gösterir. Bu özellik yüksekliği olan bir **blok nesnesi** için çalışır.

  overflow değerleri;

  - ***overflow: visible;*** **İçerik gözükür (varsayılan)**
  - ***overflow: hidden;*****İçerikten taşan kısım kırpılır.**
  - ***overflow: scroll;*** **İçeriğin taşan kısmı görünmez ama kaydırma çubuğu ile gözükür**
  - ***overflow: auto;*** **İçeriğin taşan kısmı varsa otomatik kaydırma çubuğu gelir.**

- **clear** özelliği**:** Nesneleri float ile hizalarken bazen bu hizalama etkilerinden kurtulmamız gerekebilir. Örneğin iki paragrafınız olsun ve bunlardan sadece ilkini float uygulanmış elementin yanına koymak istiyorsunuz, bu durumda ikinci paragrafa clear özelliği atadığınızda bu element float uygulanmış elementin altında konumlanacaktır.

  clear değerleri:

  - ***clear: none;*** **Clear özelliği uygulanmaz.**
  - ***clear: left;*** **Kaydırma etkisi "sol" yönünde olmaya devam eder.**
  - ***clear: right;*** **Kaydırma etkisi "sağa" olur.**
  - ***clear: both;*** **float etkisi yani kaydırma etkisi kaybolur.**

NESNELERİN GÖRÜNÜRLÜĞÜ

- Nesnelerin görünürlük özellikleri için **display** özelliğini kullanmamız gerekir.

  display değerleri:

  - ***display: none;*** **Nesne görünmez ve sayfada yokmuş gibi davranır.**
    - ***visibility: hidden;*** **Nesne görünmez ancak sayfada varmış davranır ve yer kaplar.**
    - ***visibility: collapse;*** **Tablodaki satır veya sütunların görünmemesini sağlar.**
  - ***display: block;*** **Nesne tüm satırı kapsayacak şekilde görünür.**
  - ***display: inline;*** **Nesne sadece olduğu yeri kapsar. Bu durumda nesne genişlik ve yükseklik gibi değerler alamaz.**
  - ***display: inline-block;*** **Nesne hem olduğu yeri kapsar hem de genişlik ve yükseklik gibi değerler alabilir.**

NESNELERİ KONUMLANDIRMA

- Nesneleri kapsamına göre konumlandırmak için **position** özelliğini kullanmamız gerekir.

  position değerleri:

  - ***position: static;*** **Nesne varsayılan konumundadır.**
  - ***position: relative;*****Nesnenin konumu neredeyse oradan konumlandırma yapar. Ör:**

- ***h2*** *{*
  position*: relative;*
  left*: -20px;*
  }
  - Yukarıdaki örnekte h2 elementi kullanıldığı yerden 20 piksel sola kayacak ve oradan başlayacaktır. Bu durumda diğer elementlerin alanına girebilecektir.
- ***position: absolute;*** Nesne akışın dışına çıkar ve sayfanın istediği yerine konumlanır. Eğer bir üst kapsayıcı nesnesi varsa ve kapsayıcı nesneye relative özelliği verilirse o zaman nesne kapsayıcı nesneye göre konumlandırılır.
  - Örneğin h2 elementinin sayfanın başlangıç noktasının (0, 0) 100 piksel sağında ve 150 piksel altında (100, 150) olmasını istiyorsak:
  - ***h2*** *{*
    position*: absolute;*
    left*: 100px;*
    top*: 150px;*
    }
- ***position: fixed;*** Bir nesneyi kaydırma çubuklarından etkilenmeden sabit bir yerde konumlandırmak için kullanırız. Örneğin sayfanın sağ alt köşesindeki canlı sohbet menüsü gibi.

  Ör: ***p****.fixed {*
  position*: fixed;*
  top*: 30px;*
  right*: 5px;*
  }
  - Yukarıdaki kod çalıştırılırsa "fixed" özelliğine sahip P elementinin sağdan 5 piksel üstten 30 piksel noktasında sabit bir şekilde kaldığını göreceğiz.
- ***position: sticky;*** sticky kaydırma çubuklarını takip eder ve istenilen pozisyon yakalandığında nesneyi o noktada sabitler. Örneğin bir nesne için yukarıdan 0 konumuna geldiğimiz anda nesnenin sabitlenmesini isteyebiliriz.
- ***z-index:*** Nesnelerin görünürlük önceliğini belirlemek için kullanılır. Nasıl Photoshop tarzı programlarda katmanlarla çalışılıyor ve katmanların sırası belirlenebiliyorsa, CSS'de de z-index özelliği ile bunu belirleyip bir HTML elementinin diğerinin üzerinde ya da altında görünmesini sağlayabiliriz. **Ör:**
  - ***img*** *{*
    position*: absolute;*
    left*: 0px;*
    top*: 0px;*
    z-index*: -1;*
    }
  - Yukarıda -1 değeri ile belirttiğimiz resim diğer tüm elementlerin altında görünecektir. Çünkü diğer elementlerde bu değer belirtilmediyse 0 (sıfır) olarak algılanacaktır. Buna karşın bu element -1 olduğu için alt katman kabul edilecektir. Negatif değerler en alt katmanı pozitif değerler üst katmanları ifade eder. Örneğin biri -2 biri -5 olan iki z-index özelliğine sahip elementten daha büyük olan -2 özelliğine sahip element daha üstte görünecektir.

NESNELERİN BİRBİRLERİNE GÖRE MESAFESİ

- Her bir nesnenin etrafında olan bir kutuyu temsil eder. Bu kutu modeli nesnelerin birbiri ile arasındaki mesafe **(margin)**, kutu ile içerik arasındaki mesafe **(padding)** ve kutunun etrafındaki kenarlık (border) ile ilgili çalışmaları temsil eder.
- **margin:** Nesnelerinin birbiri ile olan mesafelerini belirtir.
  - ***margin: top right bottom left;*** **4 köşeden margin değerlerini tek tek belirleyebiliriz.**
  - ***margin: top-bottom right-left;*** **2 değer şeklinde bir kullanım yapabiliriz.**
  - ***margin: top-right-bottom-left;*** **4 köşe için tek bir değer kullanımı da yapabiliriz.**
  - ***margin: auto;*** **otomatik hizalama ile nesneler ortalanır.**
  - Örnekler:

    div { margin:10px 20px; }

    **** üst-alt için 10px ve sağ-sol için ise 20px' lik bir boşluk bırakmış oluruz.**

    div { margin:10px; }

    **** 4 köşe için 10px margin değeri vermiş oluruz.**

- **padding:** Nesnelerin içeriği ile kenarlığı arasındaki dolgu genişliğini belirler. margin' de olduğu gibi 4 köşeden padding kullanımı yapabiliriz.
- **box-sizing:** Bir nesnenin kapladığı alan nesnenin genişliği **+ padding + border** değerlerinin toplamıdır. Ancak ***box-sizing: border-box*** dediğimizde bu hesaba padding ve border katılmayacaktır. Nesnenin kapladığı toplam alan width ya da height ile belirlenecektir.