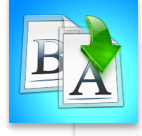


CSS'TE KUTU MODELİ, KALITSALLIK VE ÖNCELİKLER



İÇİNDEKİLER

- CSS'te Kutu Modeli
 - Div
 - Float ve clear
 - Position
 - Z-index
- CSS'te Öncelikler
- CSS'te Kalıtsallık



HEDEFLER

- Bu üniteyi çalıştıktan sonra;
 - CSS'te kutu modelinin ne olduğunu tanımlayabilecek,
 - <div> etiketi ile stil tanımlarını ve sonucunu gösterebilecek,
 - Float ve clear kullanan stil tanımlarını ve sonucunu gösterebilecek,
 - Position ile ilgili stil tanımlarını ve sonuçlarını seçebilecek,
 - CSS'te öncelikler ile ilgili tanımları ve sonuçlarını belirleyebilecek ve
 - CSS'te kalıtsallıklarla ilgili tanımları ve sonucunu gösterebileceksiniz.

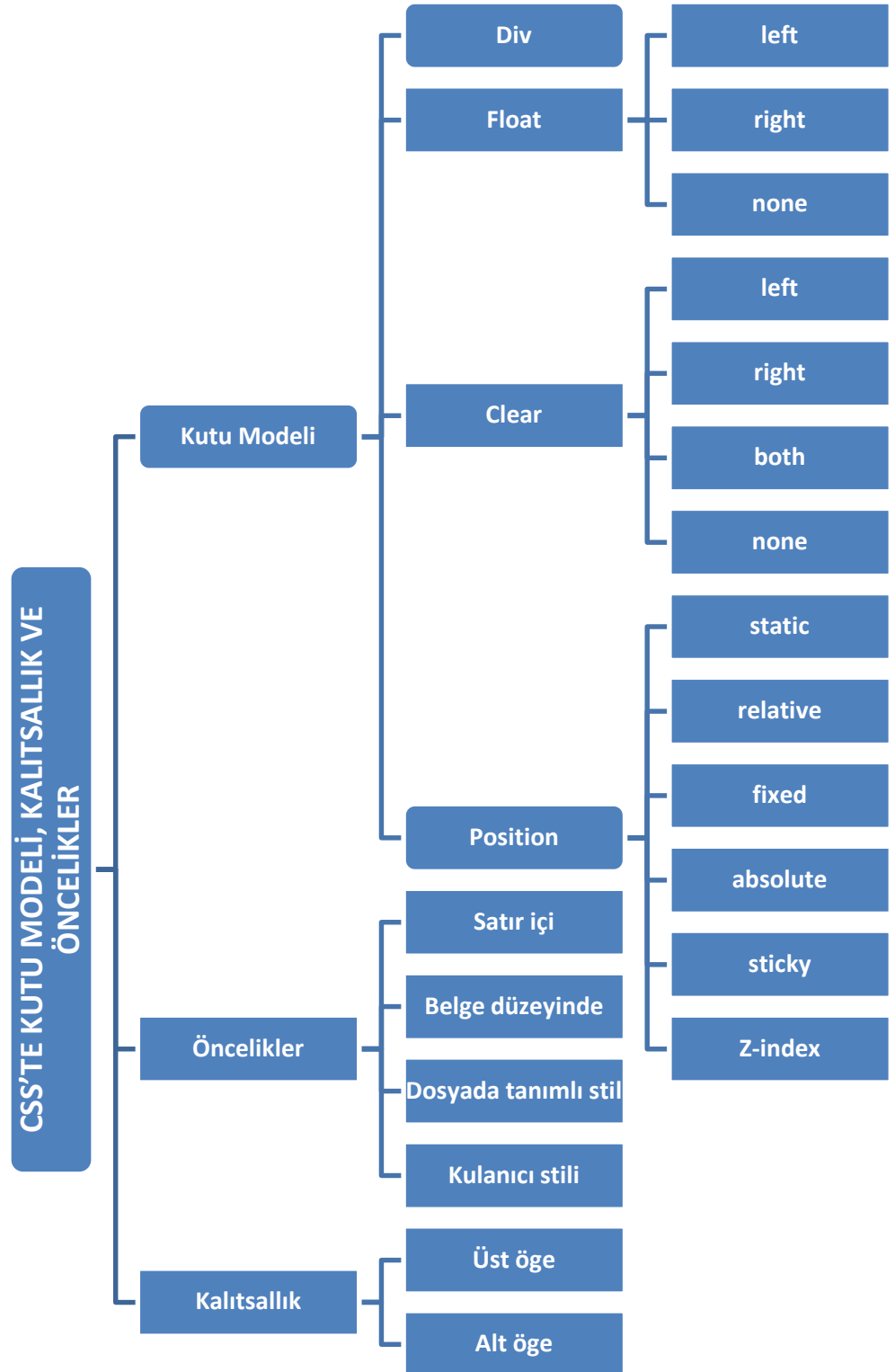


Atatürk Üniversitesi
Açıköğretim Fakültesi

İNTERNET PROGRAMCILIĞI I Doç. Dr. Ercan TOP

ÜNİTE

8



GİRİŞ

CSS ile ilgili bölümlere CSS (Cascading Style Sheets- Basamaklı Stil Sayfaları) kavramından bahsedilerek başlanmıştır. CSS ile ilgili ilk bölümde stil sayfalarının oluşturulduğu kural yapıları, CSS sürümleri, stil tanımlama yöntemleri, stil tanımında kullanılan seçici ve bildirim bloku ve seçici türleri hakkında detaylı bilgiler verilmiştir. Bölümün geri kalanında ise metin düzenlemeyle ilgili sık kullanılan özellikler ve bu özelliklere atanabilecek değerlerden bahsedilmiştir.

CSS ile ilgili ikinci bölümde ise öncelikle web sayfalarında renk kavramı hakkında bilgiler verilip color, background ve HSLA özelliklerinden ve tanımlanma yöntemlerinden bahsedilmiştir. Span özelliği ile ilgili bilgiler ve kullanımından örnekler verilmiştir. Sonrasında ise tabloların stillerinin düzenlenmesi ile ilgili olarak border, border-collapse, height, width, padding, margin, table color, hover ve nth-child özelliğinden bahsedilmiştir.

CSS ile ilgili bir sonraki bölümde ise liste ile ilgili özelliklerin (list-style-type, list-style-image, list-style position, li:hover) ve görüntülerle ilgili özelliklerin (border, border-image, border-radius, opacity, width, height, background-image) yaygın olarak kullanımları incelenmiş ve kullanımları hakkında çeşitli örnekler gösterilmiştir.

Bu bölümde ise öncelikle CSS kutu modelinden bahsedilecektir. Kutu modeli ile ilgili öncelikli olarak <div> etiketi işlenecektir. Sonrasında ise kutu modeli kavramının kullanımını kolaylaştıran float, clear ve position özellikleri incelenecektir. Position (konum) özelliğinin alabileceği değerler olan static, relative, fixed, absolute, sticky detaylı olarak irdelenecektir. Bölümün son kısmında ise CSS'te öncelikler ve kalıtsallık konusu hakkında bilgiler verilecektir. Bölüm boyunca özelliklerin kullanım şekilleri ve yöntemleri hakkında detaylı açıklamalar yapılacak, kullanım şekilleri ve yöntemleri ile ilgili tam HTML dosyaları veya HTML dosyalarının bazı bölümlerinin ekran görüntüleri paylaşılacaktır. Hazırlanan HTML kodlarının tarayıcılarda açıldığında ortaya çıkan görüntüleri de HTML dosyalarının içerikleriyle birlikte verilerek, öğrencilerin HTML kodu ve bu kodlarla ortaya çıkacak görüntüyü zihinlerinde eşleştirmelerine yardımcı olunmaya çalışılacaktır.



Kutu modeli, HTML öğelerinin etrafına kenarlık eklenmesine ve elemanlar arasında boşluk tanımlanmasına olanak sağlar.

CSS KUTU MODELİ

CSS'te, tasarım ve düzen hakkında konuşulurken "*kutu modeli*" terimi kullanılmaktadır. Tüm HTML öğeleri kutular olarak kabul edilebilir. Kutu modeli, HTML öğelerinin etrafına kenarlık eklenmesine ve elemanlar arasında boşluk tanımlanmasına olanak sağlar. *Kutu modeli kenar boşlukları, kenarlıklar, dolgular ve içerikten oluşmaktadır*. Şekil 8.1'de kutu modeli gösterilmektedir. Kutu modelindeki öğeleri incelersek;

- *İçerik*- Metin ve görüntülerin de olduğu her türlü kutu içeriğidir.
- *Dolgular*- İçeriğin etrafındaki alanı belirler. Dolgular şeffaftır.

- *Kenarlıklar*- Dolgu ve içeriğin etrafından geçen çizgilerdir. Kenarlıkların kalınlık ve renk özelliği tanımlanabilmektedir.
- *Kenar boşlukları*- Ögenin sınırının dışındaki alanı belirlemek için kullanılır. Kenar boşlukları da dolgular gibi şeffaftır.



Şekil 8.1. Kutu Modeli (Box Model)



Bir ögenin genişlik ve yükseklik özellikleri CSS ile ayarlandığında, yalnızca içerik alanının genişliği ve yüksekliği ayarlanmış olur.

Bir ögenin genişlik ve yükseklik özellikleri CSS ile ayarlandığında, *yalnızca içerik alanının genişliği ve yüksekliği* ayarlanmış olur. Bir ögenin tam boyutunu hesaplamak için; *dolgu, kenarlıklar ve kenar boşluklarının* da eklenmesi gerekmektedir. Örneğin Tablo 8.1.'deki gibi tanımlanan bir <div> etiketinin genişliği ve yüksekliğini hesaplamak için yükseklik ve genişlik değeri ile dolgu (padding), kenar boşlukları (margin) ve kenarlık değerleri de hesaplanmalıdır. Bu durumda <div> etiketinin genişliği:

- 300px (width)
- 2 * 0px (sağ + sol dolgu)
- 2 * 5px (sağ + sol kenar)
- 2 * 10px (sağ + sol kenar boşluğu)
- =330px olmaktadır.

<div> etiketinin yüksekliği:

- 400px (height)
- 2 * 0px (üst + alt dolgu)
- 2 * 5px (üst + alt kenar)
- 2 * 10px (üst + alt kenar boşluğu)
- =430px olmaktadır.

Tablo 8.1. Tam Boyut Örneği

Html Sayfası
<pre><style type="text/css"> div{height:400px; width:350px; margin:10px; padding:0px; border:5px solid green;} </style></pre>

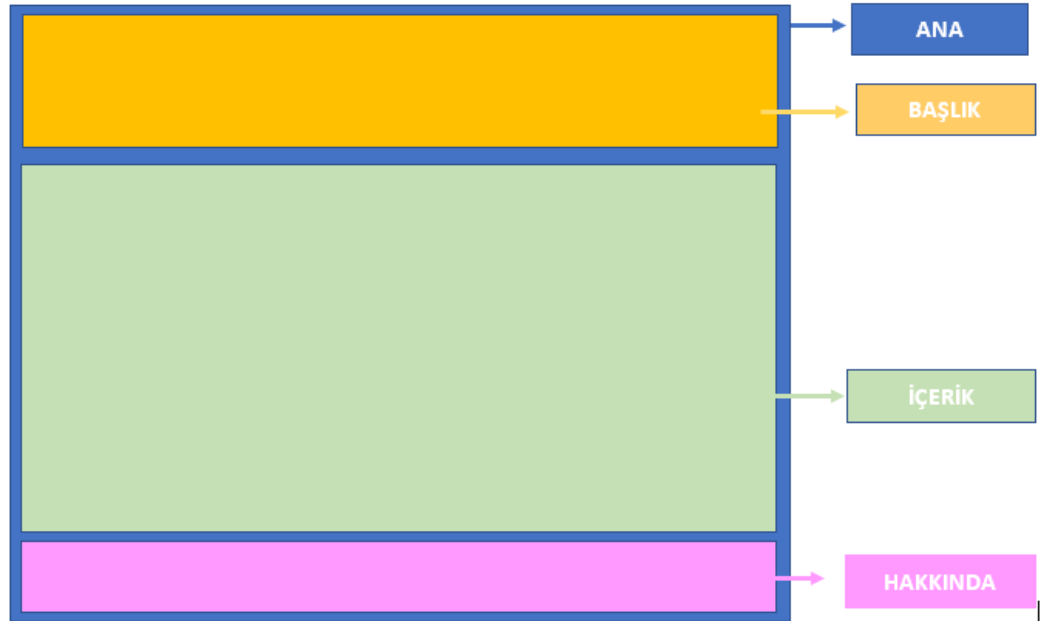
Div



HTML'deki kutu modeli teriminin iyi bir örneğini teşkil etmektedir.

<div> etiketi bir web sayfasını düzenlemek için CSS ile birlikte çok sık kullanılmaktadır. HTML'deki *kutu modeli teriminin iyi bir örneğini* teşkil etmektedir. Bir <div> etiketi, HTML belgesindeki bir bölümü veya bölmeyi tanımlamak için kullanılmaktadır. <div> etiketi genellikle diğer HTML öğelerinin CSS kullanılarak stillendirilmesi ile belirli görevleri gerçekleştirmek için bir kapsayıcı olarak kullanılmaktadır. *Varsayılan olarak, tarayıcılar her zaman <div> ögesinden önce ve sonra bir satır sonu vermektedir.* Ancak bu değerler CSS tanımlanarak değiştirilebilirler.

Şekil 8.2.'de örnek olarak hazırlanan bir sayfanın yerleşim şeması gösterilmektedir. Sayfa içerisinde üç farklı bölme olan (BAŞLIK, MENÜ ve İÇERİK) ANA adlı bir bölmenin içerisine yerleştirilmiştir. Daha detaylı olarak ifade etmek gerekirse ANA bölmesinin üst bölümüne BAŞLIK bölümü yerleştirilmiştir. ANA bölmesinin içinde, BAŞLIK bölümünün altında İÇERİK bölümü yerleştirilmiştir. Yine ANA bölmesinin içinde, BAŞLIK ve İÇERİK bölümünün altında HAKKINDA bölümü yerleştirilmiştir. Tablo 8.2.'de ayrıca gösterilen sayfa yerleşimini elde etmek için kullanılan stil tanımlamaları da gösterilmektedir. Bölmelerin ayırt edilmesini kolaylaştırmak için her bir bölmenin arka plan rengi farklı renklerle tanımlanmıştır. Stil tanımlaması yapılırken her bir bölme için benzer adlarla class tanımlamaları yapılmış ve sonra her bir <div> etiketine bu class tanımlamaları atanmıştır.



Şekil 8.2. Sayfa Yerleşimi Örneği

Tablo 8.2. Sayfa Yerleşimi HTML Kodu

Html Sayfası
<pre><html> <head> <meta charset="UTF-8"> <title>Stiller</title></pre>

```
<style type="text/css">
.ana{width:800px; height:600px; border:1px solid black;
background-color:#4472C4;}
.baslik {width:780px; height:100px; border:1px solid black;
margin:10px; background-color:#FFC000;}
.icerik {width:780px; height:400px; border:1px solid green;
margin:10px; background-color:#C5E0B4;}
.hakkinda {width:780px; height:50px; border:1px solid black;
margin:10px; background-color:#FF99FF;}</style>
</head>
<body>
<div class="ana">
<div class="baslik"></div>
<div class="icerik"></div>
<div class="hakkinda"></div>
</div>
</body>
</html>
```

Float ve Clear



CSS float özelliği, bir ögenin ekranda nasıl durması gerektiğini belirtmek için kullanılır.

CSS float özelliği, bir ögenin ekranda nasıl durması gerektiğini belirtmek için kullanılır. float özelliği, örneğin içeriği yerleştirmek ve biçimlendirmek için kullanılabilir. “float” blok seviye elemanlarını *üst üste yerleştirmek yerine yan yana yerleştirmeye* olanak sağlar. Kenar çubukları, çok sütunlu sayfalar, ızgaralar ve bir görüntünün etrafında akan metin içeren dergi stili makaleler dâhil olmak üzere her türlü sayfa tasarımını oluşturmaya olanak sağlarlar. Float özelliği aşağıdaki değerlerden birine sahip olabilir:

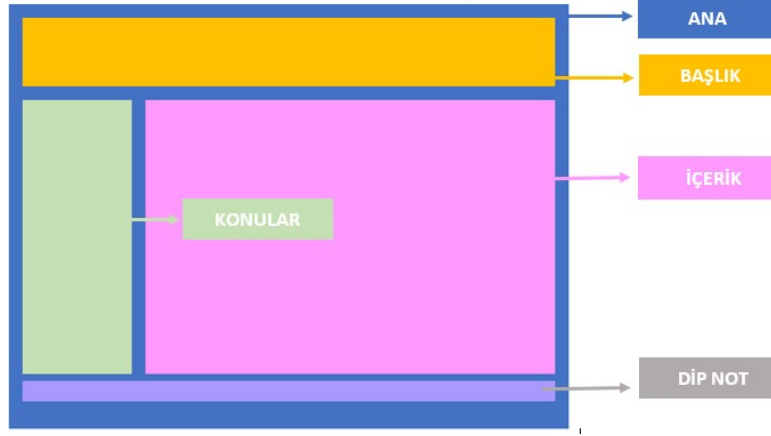
- *left*: Eleman içinde bulunduğu kabının soluna kayar.
- *right*: Eleman içinde bulunduğu kabının sağına kayar.
- *none*: Öge değişmez (yalnızca metinde gerçekleştiği yerde görüntülenir).

CSS clear özelliği, hangi öğelerin temizlenen ögenin yanında ve hangi tarafında kalacağını belirlemek için kullanılmaktadır. Clear özelliği, aşağıdaki değerlerden birine sahip olabilir:

- *none*- Her iki tarafta da elemanlar bulunmasına izin verir. Bu varsayılan değerdir.
- *left*- Sol tarafta eleman bulunmasına izin verilmez.
- *right*: Sağ tarafta eleman bulunmasına izin verilmez.
- *both*- Sol veya sağ tarafta eleman bulunmasına izin verilmez.

Clear özelliği *genelde bir öğede float özelliği* ile kullanılmaktadır. Şekil 8.3.’te “float” ve “clear” özelliklerinin birlikte kullanıldığı bir sayfanın yerleşim şeması gösterilmektedir. Sayfa içerisinde dört farklı bölme olan (BAŞLIK, KONULAR, İÇERİK VE DİPNOT) ANA adlı bir bölmenin içerisine yerleştirilmiştir. Daha detaylı olarak ifade etmek gerekirse ANA bölmesinin üst bölümüne BAŞLIK bölümü yerleştirilmiştir. ANA bölmesinin içinde, BAŞLIK bölümünün altında ve sol tarafta KONULAR bölümü yerleştirilmiştir. Yine BAŞLIK bölümünün altında ve KONULAR

bölmesinin sağına İÇERİK bölümü yerleştirilmiştir. KONULAR ve İÇERİK bölmelerinin altında iki bölmenin genişliğinde tek parça olarak DİPNOT bölümü yerleştirilmiştir. Tablo 8.3.'te ayrıca gösterilen sayfa yerleşimini elde etmek için kullanılan stil tanımlamaları da gösterilmektedir. Bölmelerin ayırt edilmesini kolaylaştırmak için her bir bölmenin arka plan rengi farklı renklerle tanımlanmıştır. Stil tanımlaması yapılırken her bir bölme için benzer adlarla class tanımlamaları yapılmış ve sonra her bir <div> etiketine bu class tanımlamaları atanmıştır.



Şekil 8.3. Sayfa Yerleşimi Örneği (float ve clear)

Tablo 8.3. Sayfa Yerleşimi Örneği (float ve clear)



Clear özelliğinin alabileceği değerler; left, right, both ve none olarak belirlenmiştir.

Html Sayfası

```
<html> <head>
  <meta charset="UTF-8">
  <title>Stiller</title>
  <style type="text/css">
    .ana{width:800px; height:600px; padding:10px; background-color:#4472C4;}
    .baslik {width:auto; height:100px; margin:10px; background-color:#FFC000;}
    .konular {width:20%; height:400px; margin:10px;
      background-color:#C5E0B4; float:left;}
    .icerik {width:75%; height:400px;
      margin:10px; background-color:#FF99FF; float:right;}
    .dipnot{width:auto; height:30px;
      margin:10px;background-color:#AA99FF; clear:both;}
  </style> </head> <body>
  <div class="ana">
    <div class="baslik"></div>
    <div class="konular"></div>
    <div class="icerik"></div>
    <div class="dipnot"></div>
  </div>
</body> </html>
```

Tablo 8.3.'te, Şekil 8.3.'teki görüntüyü elde etmek için kullanılan dosyanın içeriği incelenebilmektedir. "ana" still class'ı kullanılan <div> etiketi diğer bütün <div> etiketlerini içerisinde barındırmaktadır. "baslik" stil class tamınında kullanılan "width:auto;" bildirimi, genişliğin ögenin içinde bulunduğu kabın

genişliğine göre ayarlanmasını otomatik olarak sağlamaktadır. “konular” class tanımında kullanılan “float:left;” bildirimi, tanımlanan ögenin bulunduğu kabın soluna yerleşmesini sağlamaktadır. “içerik” class tanımında kullanılan “float:right;” bildirimi, tanımlanan ögenin bulunduğu kabın sağına yerleşmesini sağlamaktadır. Şekilde “içerik” class ataması yapılan <div> etiketi, “konular” class ataması yapılan <div> etiketinin sağına yerleşmiştir. Eğer “içerik” bölümünün genişliği “konular” bölümünün sağında kalan alandan büyük olsaydı, “içerik” bölümü konular bölümünün altında ve sağ tarafa yaslanmış olarak yerleşecekti. “dipnot” class tanımında kullanılan “clear:both;” bildirimi, bu class tanımını kullanan <div> etiketi içeriğinin “konular” ve “içerik” bölümünün altına yerleşmesini sağlamıştır. Eğer “clear:both;” bildirimi yapılmamış olsaydı, Şekil 8.4.’teki görüldüğü gibi “dipnot” bölümü “başlık” bölümünün altına ve “konular” ve “içerik” bölmelerinin arkasına yerleştirilmiş olacaktı. Bir başka ifade ile “float” bildirimi kullanılan css tanımları normal sıralamanın değişmesine neden olabilmektedir, bu yüzden dikkatli olmak gerekmektedir.



Şekil 8.4. Sayfa Yerleşimi Örneği (clear:both Kullanılmadan Önceki Görünüm)

Position

Position özelliği, bir öge için kullanılan konumlandırma yönteminin türünü belirtmektedir. Özellik, bir içeriğin sayfada nerede görüntüleneceğini tanımlamak için kullanılmaktadır. Ögeler *top (üst)*, *right (sağ)*, *bottom (alt)*, *left (sol)* ve *z-index (z-indeks)* değerleri kullanılarak konumlandırılır. Ancak “position” özelliği **öncelikle ayarlanmadıkça** bu değerler çalışmaz. Ayrıca değerler, “position” özelliğine bağlı olarak farklı çalışırlar. Position özelliği için beş seçenek tanımlanabilmektedir:

- *static*,
- *relative*,
- *fixed*,
- *absolute* ve
- *sticky*.

z-indeksi nedir?

Yükseklik ve genişlikten sonra (x, y) üçüncü boyuttur. Bir öge, z indeks değeri arttıkça diğer öğelerin önüne gelmektedir.

Static

HTML öğeleri varsayılan şekilde statik olarak konumlandırılmıştır. Statik konumlandırılmış öğeler; *üst*, *alt*, *sol* ve *sağ değerlerinden* etkilenmezler. Position



Position özelliğinin alabileceği değerler; static, relative, fixed, absolute ve sticky olarak belirlenmiştir.



Position değeri “static” olan bir öge, sayfanın normal akışına göre yerleştirilir.

değeri “static” bir ögenin içeriği, farklı şekilde konumlandırılmaz, içerik sayfanın normal akışına göre yerleştirilir. *z-index değeri de “static” seçeneği için bulunmamaktadır.* Tablo 8.4.’te “static” seçeneğine örnek bir class tanımlaması gösterilmektedir. Farklı renklerde, genişliği ve yüksekliği 100 piksel olan iki tane class tanımlaması yapılmış, “statik” adlı class tanımında ayrıca position özelliği için “static” değeri tanımlanmıştır. Dosyanın tarayıcıda açılmış hâlinde olduğu gibi herhangi bir “position” bildirimi yapılmamış olsaydı da sayfanın tarayıcıda görünümü aynı olacaktı.

Tablo 8.4. Static Konum Yerleşimi Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> div.kontrol { background-color:#EE00FF; height:100px; width:100px;} div.statik { position: static; background-color:#C5E0B4; height:100px; width:100px;} </style> </head> <body> <div class="kontrol"></div> <div class="statik"></div> </body> </html> </pre>	

Relative




Position değeri “relative” olan bir öge, sayfanın normal akışındaki konumuna göre ögeyi yerleştirmek için kullanılmaktadır.

Relative seçeneği, sayfanın normal akışındaki konumuna göre ögeyi yerleştirmek için kullanılmaktadır. Relative ve statik olmayan tüm konum değerleri için *ögenin varsayılan konumu* değiştirilmektedir. Fakat sadece *position özelliği için relative değerini bildirmek yeterli değildir*, aynı zamanda elementin koordinatlarının yardımcı özelliklerle de ayarlanması gerekmektedir. Tablo 8.4.’teki örnekteki ikinci kareyi ilk karenin yanına almak için gerekli stil tanımı Tablo 8.5.’teki gibi yapılabilmektedir. İkinci kare, normal akışta ilk karenin hemen altına yerleşecekti. İkinci karenin, normal akıştaki yerinden ilk karenin hemen sağına gelebilmesi 100 piksel yukarı çıkması gerekmektedir, bu yüzden “top:-100px;” bildirimi yapılmıştır. İkinci kare aynı zamanda normal akışına göre soldan 100 piksel uzağa yerleştirilmektedir, normal akıştaki bulunduğu konumdan 100 piksel uzağa yerleşimi sağlamak için “left:100px;” bildirimi yapılmıştır.

Tablo 8.5. Relative Konum Yerleşimi Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
--------------	---------------------------------

<pre> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> div.kontrol { background-color:#EE00FF; height:100px; width:100px;} div.goreceli { position: relative; background-color:#C5E0B4; height:100px; width:100px; top:-100px; left:100px;} </style> </head> <body> <div class="kontrol"></div> <div class="goreceli"></div> </body> </html> </pre>	
--	--

Fixed

Fixed seçeneği; sayfa aşağı, yukarı veya yanlara doğru kaydırıldığında içeriğin *her zaman görünür* olmasını sağlamak için kullanılmaktadır. Belirtilen öğeyi *görünüm alanında sabitleştirir*, sayfayı kaydırmak, sabit konumlandırılmış öğeyi etkilemez. Sabit bir öğe, sayfada normalde yerleştirileceği yerde (içerik akışında olması gereken yerde) *herhangi bir boşluk* bırakmaz. Şekilde “sabit” class tanımında konum değeri için “position:fixed;” ve konum olarak “top:0px;” ve “right:100px;” bildirimleri yapılmıştır. Bu tanımlanan öğeyi Tablo 8.6.’daki gibi sayfanın en üstüne ve sağdan 100 piksel mesafeye yerleştirmek demektir.




Position değeri “fixed” olan bir öğe, görünüm alanında sabit yerleştirilir, sayfayı kaydırmak, sabit konumlandırılmış öğeyi etkilemez.

```

<style type="text/css">
  div.kontrol{ background-color:#EE00FF;
    height:100px; width:100px;}
  div.sabit{position:fixed; background-color:#C5E0B4;
    height:100px; width:100px; top:0px; right:100px;}
</style>

```

Tablo 8.6. Fixed Konum Yerleşimi Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> div.kontrol { background- color:#EE00FF; height:100px; width:100px;} div.sabit { position: fixed; </pre>	

<pre>background-color:#C5E0B4; height:100px; width:100px; top:0px; right:100px;} </style> </head><body> <div class="kontrol"></div> <div class="sabit"></div> </body> </html></pre>	
---	--

Absolute



Position değeri
"absolute" olan bir öge,
üst ögeye göre
konumlandırılmaktadır.

Absolute seçeneğindeki içerik, *üst ögeye göre* konumlandırılmaktadır. Üst ögenin *başlangıç noktasına* (sol üst köşe) göre otomatik olarak yerleştirilmektedir. Herhangi bir üst ögeye sahip değilse, ilk belge (<html>) üst ögesi olarak kabul edilmektedir. "absolute" olarak tanımlanan ögeler *normal belge akışından kaldırılır*. "absolute" olarak tanımlanan öge, belge akışından tamamen kaldırdığından dolayı, diğer ögeler bu durumdan etkilenmekte ve öge web sayfasından tamamen kaldırılmış gibi işleme devam edilmektedir. Tablo 8.7.'de "position:absolute;" olarak tanımlanan öge için konum olarak "top:100px; left:100px;" bildirimleri de yapılmıştır. Bu bildirim ile önceki ögeye göre üst bölümden 100 piksel ve soldan 100 piksel mesafeye yeni ögenin ("mutlak" class tanımlaması atanan <div> etiketi) şekildeki gibi yerleştirildiği görülmektedir.

Tablo 8.7. Absolute Konum Yerleşimi Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><html><head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> div.kontrol { background-color:#EE00FF; height:100px; width:100px;} div.mutlak{ position: absolute; background-color:#C5E0B4; height:100px; width:100px; top:100px; left:100px;} </style> </head> <body> <div class="kontrol"></div> <div class="mutlak"></div> </body> </html></pre>	

Sticky

sticky kullanımı ile *kullanıcının kaydırma konumuna göre* konumlandırma yapılabilmektedir. "Sticky" olarak tanımlanan öge, kaydırma konumuna bağlı olarak *göreceli (relative)* ve *sabit (fixed)* değer arasında geçiş yapabilmektedir. Oluşturulan öge kaydırma çubuğu kendi *konumuna gelene kadar "relative"*

(göreceli) değeriindeki gibi konumlandırılır, daha *sonra değer* “fixed” olarak tanımlanmış gibi sayfada gösterilmektedir. Birçok tarayıcı sürümü desteklemediğinden dolayı “sticky” özelliğini kullanırken dikkatli olmakta fayda vardır.

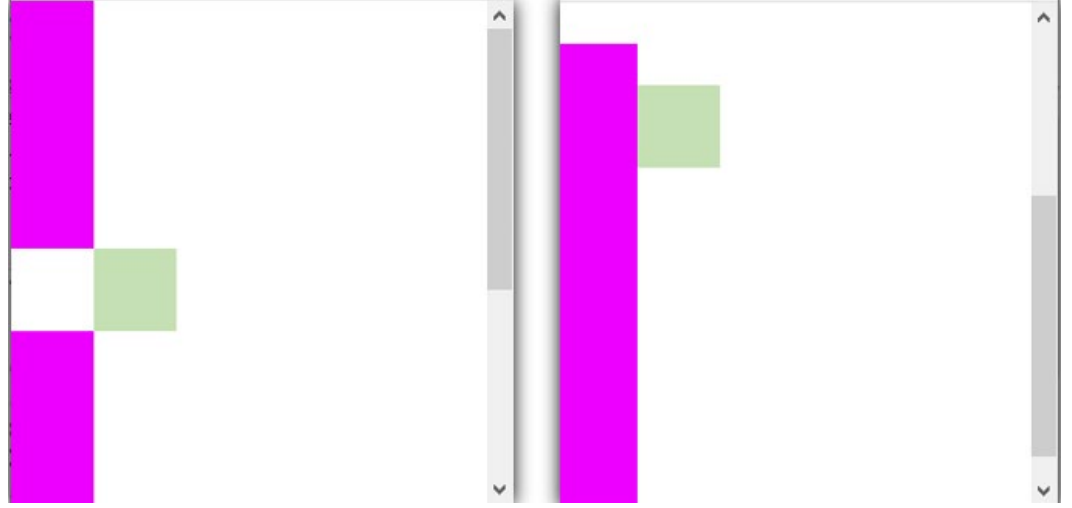
Tablo 8.8.’de “kontrol” ve “yapiskan” adlı iki class stil tanımı yapılmıştır. Özelliğin tarayıcıda daha iyi görüntülenebilmesi için “kontrol” class tanımında yükseklik değeri 300 piksel olarak tanımlanmış ve dosya içeriğinde üç defa <div> etiketine uygulanmıştır. “yapiskan” class tanımında “position:sticky;” bildirimi yapılmış, tanımlanan öge için konum olarak “top:100px; left:100px;” bildirimleri de yapılmıştır. “yapiskan” class tanımı <div> etiketine uygulanan ilk “kontrol” uygulanmasından sonra ikinci <div> etiketine uygulanmıştır. Şekil 8.5.’in solunda görüldüğü gibi sayfa tarayıcıda açıldığında “yapiskan” stil tanımı uygulanan div etiketi “position: relative;” konum bildirimindeki gibi normal akışına göre soldan 100 piksel uzağa yerleştirilmiştir. Şekil 8.5.’in sağında görüldüğü gibi kaydırma çubuğu kendi konumuna gelince, “position: fixed” konum bildirimindeki gibi “yapiskan” stil tanımı uygulanan div etiketi sayfanın köşesinden 100 piksel solda ve altta yerleştirilmiştir.

Tablo 8.8. Sticky Konum Yerleşimi Örneği

Html Sayfası
<pre> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style type="text/css"> body {padding: 0px; margin: 0px;} div.kontrol { background-color:#EE00FF; height:300px; width:100px;} div.yapiskan { position: sticky; background-color:#C5E0B4; height:100px; width:100px; top:100px; left:100px;} </style> </head> <body> <div class="kontrol"></div> <div class="yapiskan"></div> <div class="kontrol"></div> <div class="kontrol"></div> </body> </html> </pre>



Position değeri “sticky” olan bir öge, kullanıcının kaydırma konumuna göre konumlandırma yapılabilmektedir.



Şekil 8.5. Sticky Konum Yerleşimi Örneği



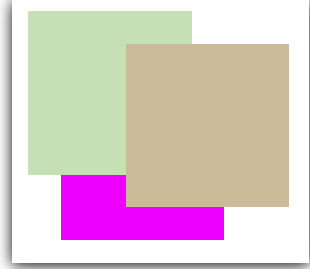
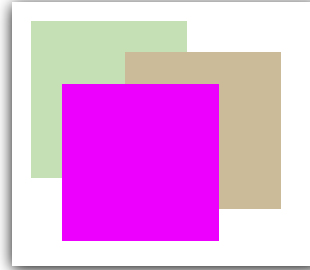
Üst üste binen içeriğin hangisinin üstte görüntüleneceğini tanımlamak için z-index özelliği kullanılmaktadır.

Z-index

Üst üste binen 2 veya daha fazla içerik olduğunda, üst üste binen içeriğin *hangisinin üstte görüntüleneceğini* tanımlamak için z-index özelliği kullanılmaktadır. z-index özelliğinin uygulanabilmesi için *öncelikle öğelere konumlandırma (position) değeri atanmış* olması gerekmektedir. Tablo 8.9.'daki stil tanımlamalarındaki z-index değerleri çıkarılırsa, dosyanın normal akışı gereği en altta "kontrol" class tanımı uygulanan <div> etiketi (mor arka plan rengi olan <div> etiketi), onun üzerinde "ikinci" class tanımı uygulanan <div> etiketi (açık yeşil arka plan rengi olan <div> etiketi) ve en üstte "ucuncu" class tanımı uygulanan <div> etiketi (açık kahverengi plan rengi olan <div> etiketi) olacaktır. z-index özelliği bütün classlar için tanımlandığında normal akış gereği en altta olması gereken "kontrol" class tanımı uygulanan <div> etiketi z-index değeri en büyük olduğu için üstte (mor arka plan rengi olan <div> etiketi), normal akış gereği en üstte olması gereken "ucuncu" class tanımı uygulanan <div> etiketi (açık kahverengi arka plan rengi olan <div> etiketi) z-index değeri ikinci büyük olduğu için ortada yer almaktadır. Ayrıca, normal akış gereği ortada olması gereken "ikinci" class tanımı uygulanan <div> etiketi (açık yeşil arka plan rengi olan <div> etiketi) z-index değeri en küçük olduğu için en altta yer almaktadır.

```
<style type="text/css">
  body {padding:0px; margin:0px;}
  div.kontrol{background-color:#EE00FF; height:100px; width:100px;
    position: relative; left:100px; top:60px; z-index:6;}
  div.ikinci{position: relative; background-color:#C5E0B4;
    height:100px; width:100px; top:-80px; left:80px; z-index:2;}
  div.ucuncu{position: relative; background-color:#CCBB99;
    height:100px; width:100px; top:-160px; left:140px; z-index:4;}
</style>
```

Tablo 8.9. Sticky Konum Yerleşimi Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre> <html> <head><title>Stiller</title> <style type="text/css"> body {padding:0px; margin:0px;} div.kontrol { background-color:#EE00FF; height:100px; width:100px; position: relative; left:100px; top:60px; z-index:6;} div.ikinci { position: relative; background- color:#C5E0B4; height:100px; width:100px; top:-80px; left:80px; z-index:2;} div.ucuncu { position: relative; background- color:#CCBB99; height:100px; width:100px; top:-160px; left:140px; z-index:4;} </style> </head> <body> <div class="kontrol"></div> <div class="ikinci"></div> <div class="ucuncu"></div> </body> </html> </pre>	<p>z-index değerleri çıkarılmış dosyanın görüntüsü</p>  <p>z-index değerleri uygulanmış dosyanın görüntüsü</p> 

CSS'DE ÖNCELİKLER



Basamaklama (cascading), CSS'de kural kümesinin adıdır.

Basamaklama (cascading), CSS'de kural kümesinin adıdır. Oluşturulan bildirimlerde *ortaya çıkan çatışmaların nasıl çözüldüğünü* belirlemektedir ve *dilin nasıl çalıştığının* temel bir parçasıdır. Tecrübeli geliştiricilerin çoğu genel bir algıya sahip olsa da bazı bölümler bazen yanlış anlaşılabilir. CSS stillerini satır içi, sayfaya gömülü ve dış stil dosyaları (CSS) olmak üzere üç şekilde tanımlanabilmektedir (Daha detaylı bilgi için beşinci bölümdeki "Css Tanımlama Yöntemleri" başlığı tekrar incelenebilir). Bunlara ek olarak sayfalarda stil kullanımını etkileyen *kullanıcı stili* ve *tarayıcı stili* olmak üzere iki tip daha bulunmaktadır. *Kullanıcı stili*nde HTML dosyasında herhangi bir stil bildirimi yapılmamışsa ve kullanıcı tarayıcıda stil tanımlamalarında değişiklik yapmışsa, tarayıcıda kullanıcının yapmış olduğu bildirimler etkin olmaktadır. *Tarayıcı stili*nde ise HTML dosyasında herhangi bir stil bildirimi yoktur ve kullanıcı da tarayıcının stil bildirimlerinde değişiklik yapmamıştır, bu dosyanın gösteriminde tarayıcı tarafından varsayılan stil ayarları kullanılmaktadır.

CSS bildiriminde kullanılan öncelik sıralarını incelemek, sayfalarda yapılan stil düzenlemelerinin istenilen şekilde görünmesini sağlamak için önemlidir.

Satır içi stil / belge düzeyinde stil



Bir html dosyasında belge düzeyinde stillere göre öncelikle baskın olan kısım satır içi stillerdir.

Satır içi stilde bir etiket, sınıf veya ID için özellikler tanımlanır. Bir html dosyasında belge düzeyinde stillere göre **öncelikle baskın olan kısım satır içi stillerdir**. Tablo 8.10.'daki örnekte <h3> etiketinin metin rengi için belge düzeyinde ve satır içi olmak üzere iki farklı şekilde stil bildirimi yapılmıştır. İlk <h3> etiketine satır içi stil uygulanmıştır. İkinci <h3> etiketi de satır içi herhangi bir bildirim olmadan bir metne uygulanmıştır. Tarayıcıda dosya açıldığında ilk <h3> etiketine satır içi stil etki etmektedir ve başlık kırmızı olarak tarayıcıda görünmektedir. İkinci <h3> etiketi uygulanan bölümde ise sayfa düzeyinde stilin (belge düzeyinde <h3> etiketi tanımında olduğu gibi metnin rengi yeşil) etkili olduğu görülmektedir.

Tablo 8.10. Satır İçi / Belge Düzeyinde Stil Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style> h3 {color:green;} </style> </head> <body> <h3 style="color: red; ">h3 etiketi + Satır içi bildirim</h3> <h3>h3 etiketi + sayfa düzeyinde</h3> </body> </html></pre>	



Bir html dosyasında sayfaya gömülü stiller, CSS dosyasındaki stillere göre baskın olan kısımdır.

Sayfaya gömülü stil / dış stil dosyaları (CSS)

Bir html dosyasında **sayfaya gömülü stiller, CSS dosyasındaki stillere göre baskın olan kısımdır**. Tablo 8.11.'deki örnekte <h3> etiketinin metin rengi için sayfaya gömülü ve dış stil dosyasında (stil.css) olmak üzere iki farklı şekilde stil bildirimi yapılmıştır. Tarayıcıda dosya açıldığında <h3> etiketine sayfaya gömülü stil etki etmektedir ve başlık yeşil olarak tarayıcıda görünmektedir.

Tablo 8.11. Belge Düzeyinde / Dış Stil Dosyaları Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre><html> <head> <meta charset="UTF-8"> <title>Stiller</title> <link rel="stylesheet"</pre>	


```
type="text/css" href="stil.css">
<style>
  h3 {color:green;}
</style>
</head>
<body>
<h3> h3 etiketi + sayfa
düzeyinde</h3>
</body>
</html>
```

“stil.css” Dosyası İçeriği

```
1 h3{color:red;};
2 p{text-align : center;};
```

Dosyada tanımlı stil olması / kullanıcı stili

Kullanıcı stilinde HTML dosyasında herhangi bir stil bildirimi yapılmamışsa ve kullanıcı tarayıcıda stil tanımlamalarında değişiklik yapmışsa, tarayıcıda *kullanıcının yapmış olduğu bildirimler* etkin olur. Bir başka ifade ile kullanıcı stili yerine, HTML dosyasında bir stil bildirimi yapılmışsa etkin olan bu stil bildirimlerdir.

Kullanıcı stili / tarayıcı stili

Tarayıcı stilinde HTML dosyasında herhangi bir stil bildirimi yoktur ve kullanıcı da tarayıcının stil bildirimlerinde değişiklik yapmamıştır, bu dosyanın gösteriminde tarayıcı tarafından *varsayılan stil ayarları* kullanılır. Bir başka ifadeyle tarayıcı stilleri HTML dosyasında herhangi bir bildirim olmadığında ve kullanıcı tarayıcının ayarlarında değişiklik yapmadığında etkin olur.

CSS'DE KALITSALLIK (INHERITANCE)

HTML öğeleri bir soyağacında olduğu gibi bir bağ ile birbirine bağlıdır. Bir başka ifade ile soy ağacının öğeleri arasında olduğu gibi HTML öğeleri arasında da bir kalıtsallık söz konusudur. Aynı aile içinde kalıtsal değerlerin büyükten küçüğe geçtiği gibi HTML öğeleri arasında da *CSS özellikleri kalıtsal olarak alt etiketleri* (çocuk etiket) etkiler. Kalıtsallık CSS kullanımında bir bildirimin tek tek bütün etiketlere uygulanması yerine *ata etikete uygulayarak alt (çocuk) etiketlere de otomatik olarak uygulanmasını* sağlar.

Bir ögenin stilleri alabilmesinin bir yolu da *mirastır (kalıttır)*. Basamaklama, sık sık kalıtım kavramıyla birleştirilmektedir. İki konu birbiriyle ilişkili olsa da aralarında farklılıklar vardır ve bu farklılıklar ayırt edilmelidir. Bir ögenin belirli bir özellik için basamaklı bir değeri yoksa, onu bir ata ögesinden (üst ögeden) miras alabilir. Örneğin <body> etiketine font-family (yazı tipi ailesi) uygulamak yaygındır. Daha sonra <body> etiketi içindeki tüm ata elemanları, bu yazı tipini devralacaktır. Bir başka ifadeyle <body> etiketinde uyguladığınız font-family değerini kullanmak için sayfadaki her bir etikete bu bildirimi yapmanız gerekmez, zaten body içindeki diğer etiketler bu değeri <body> etiketinden miras alırlar.



Kalıtsal değerlerin büyükten küçüğe geçtiği gibi HTML öğeleri arasında CSS özellikleri kalıtsal olarak alt etiketleri etkiler.



Margin (kenar), padding (dolgulama), background (arka plan rengi), width (genişlik) gibi bazı özelliklerde kalıtsallık yoktur.

Ancak CSS'de *bütün özellikler miras alınmaz*, margin (kenar), padding (dolgulama), background (arka plan rengi), width (genişlik) gibi bazı özelliklerde kalıtsallık yoktur.

Tablo 8.12.'de HTML dosyasında belge düzeyinde <body> etiketi için stil tanımlanmıştır. <body> etiketi için font-family (yazı tipi ailesi) ve color (metin rengi) özellikleri için bildirim yapılmıştır. HTML dosyasının içeriğinde ise sırayla <h3> etiketi, <p> etiketi ve satır içi stil tanımlama içeren <p> etiketi kullanılmıştır. Satır içi stil bildirimi içeren <p> etiketinde color ve font-weight özelliği için ayrı ayrı bildirimler yapılmıştır.

Şekilde HTML dosyasının tarayıcıda açıldığında ekran görüntüsü de gösterilmiştir.

- Ekran görüntüsünde HTML dosyasında <h3> etiketi ile ilgili herhangi bir CSS tanımlaması olmamasına rağmen, <h3> etiketi içindeki metnin <body> etiketinde tanımlandığı gibi font-family ve color özelliklerini miras aldığı görülmektedir.
- İlk <p> etiketinden önce de <p> etiketi ile ilgili herhangi bir CSS tanımı olmadığı için <p> etiketi de font-family ve color özelliklerini <body> etiketinden miras almıştır.
- İkinci <p> etiketi içinde satır içi olarak öncelikle color özelliği için bildirim yapılmıştır. HTML dosyasının ekran görüntüsü incelendiğinde <p> etiketi içindeki metnin <body> etiketindeki font-family özelliğini miras aldığı görülmektedir. Bir başka ifade ile ikinci <p> etiketinde bir font-family özelliği için bildirim yoktur. <body> etiketinde bir bildirim olduğu için bu özellik oradan miras kalmıştır. <p> etiketinde color değeri olarak satır içi yapılan color değerinin uygulandığı (basamaklama), <body> etiketi için yapılan color bildiriminin miras alınmadığı görülmektedir. İkinci <p> etiketinde font-weight özelliği için de bildirim yapılmıştır ve bu özellik için başka bir yerde bildirim yoktur. Dolayısı ile satır içi olarak font-weight özelliğine yapılan bildirimin metne doğrudan uygulandığı görülmektedir.

Tablo 8.12. CSS'te Kalıtsallık Örneği

Html Sayfası	Sayfanın Tarayıcıdaki Görüntüsü
<pre> <html> <head> <meta charset="UTF-8"> <title>Stiller</title> <style> body { font-family: Verdana Arial; color: green; } </style> </head> <body> <h3> h3 etiketi</h3> <p> p etiketi </p> <p style="color: black; font-weight: </pre>	

```
bold; ">p etiketi + inline stil tanımı  
</p>  
</body>  
</html>
```



**Bireysel
Etkinlik**

- Stillerin miras olarak kullanılması ne tür sorunlara yol açabilir?
- Bu sorunları gidermek için ne tür önlemler alınabilir?



Özet

•CSS KUTU MODELİ

•CSS'te, tasarım ve düzen hakkında konuşulurken "kutu modeli" terimi kullanılmaktadır. Kutu modeli, HTML öğelerinin etrafına kenarlık eklenmesine ve elemanlar arasında boşluk tanımlanmasına olanak sağlar. Kutu modeli kenar boşlukları, kenarlıklar, dolgular ve içerikten oluşmaktadır. Bir ögenin tam boyutunu hesaplamak için, dolgu, kenarlıklar ve kenar boşluklarının da eklenmesi gerekmektedir.

•**Div:** <div> etiketi bir web sayfasını düzenlemek için CSS ile birlikte çok sık kullanılmaktadır. HTML'deki kutu modeli teriminin iyi bir örneğini teşkil etmektedir. <div> etiketi genellikle diğer HTML öğelerinin CSS kullanılarak stillendirilmesi ile belirli görevleri gerçekleştirmek için bir kapsayıcı olarak kullanılmaktadır.

•**Float ve Clear:** CSS float özelliği, bir ögenin ekranda nasıl durması gerektiğini belirtmek için kullanılır. CSS clear özelliği, hangi öğelerin temizlenen ögenin yanında ve hangi tarafında kalacağını belirlemek için kullanılmaktadır. Clear özelliği genelde bir ögede float özelliği ile kullanılmaktadır.

•**Position:** Position özelliği, bir öge için kullanılan konumlandırma yönteminin türünü belirtmektedir. Özellik, bir içeriğin sayfada nerede görüntüleneceğini tanımlamak için kullanılmaktadır. Öğeler top (üst), right (sağ), bottom (alt), left (sol) ve z-index (z-indeks) değerleri kullanılarak konumlandırılır. Değerler, "position" özelliğine bağlı olarak farklı çalışırlar.

•**Static:** HTML öğeleri varsayılan olarak statik olarak konumlandırılmıştır. Statik konumlandırılmış öğeler, üst, alt, sol ve sağ değerlerinden etkilenmezler.

•**Relative:** relative seçeneği, sayfanın normal akışındaki konumuna göre ögeyi yerleştirmek için kullanılmaktadır.

•**Fixed:** Fixed seçeneği, sayfa aşağı, yukarı veya yanlara doğru kaydırıldığında içeriğin her zaman görünür olmasını sağlamak için kullanılmaktadır.

•**Absolute:** absolute seçeneğindeki içerik, üst ögeye göre konumlandırılmaktadır. Üst ögenin başlangıç noktasına (sol üst köşe) göre otomatik olarak yerleştirilmektedir.

•**Sticky:** sticky kullanımı ile kullanıcının kaydırma konumuna göre konumlandırma yapılabilmektedir. "Sticky" olarak tanımlanan öge, kaydırma konumuna bağlı olarak göreceli (relative) ve sabit (fixed) değer arasında geçiş yapabilmektedir. Oluşturulan öge kaydırma çubuğu kendi konumuna gelene kadar "relative" (göreceli) değerindeki gibi konumlandırılır, daha sonra değer "fixed" olarak tanımlanmış gibi sayfada gösterilmektedir.

•**Z-index:** Üst üste binen 2 veya daha fazla içerik olduğunda, üst üste binen içeriğin hangisinin üstte görüntüleneceğini tanımlamak için z-index özelliği kullanılmaktadır.

•CSS'DE ÖNCELİKLER

•Basamaklama (cascading), CSS'de kural kümesinin adıdır. Oluşturulan bildirimlerde ortaya çıkan çatışmaların nasıl çözüldüğünü belirlemektedir ve dilin nasıl çalıştığının temel bir parçasıdır. CSS bildiriminde kullanılan öncelik sıralarını incelemek, sayfalarda yapılan stil düzenlemelerinin istenilen şekilde görünmesini sağlamak için önemlidir.

•**Satır içi stil / belge düzeyinde stil:** Satır içi stilde bir etiket, sınıf veya ID için özellikler tanımlanır. Bir html dosyasında belge düzeyinde stillere göre öncelikle baskın olan kısım satır içi stillerdir.



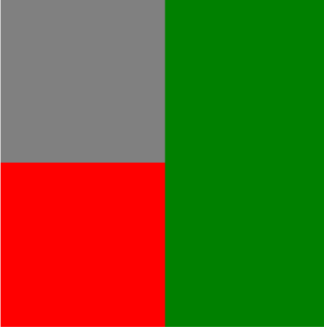
Özet(devamı)

- **Sayfaya gömülü stil / dış stil dosyaları (CSS):** Bir html dosyasında sayfaya gömülü stiller, CSS dosyasındaki stillere göre baskın olan kısımdır.
- **Dosyada tanımlı stil olması / kullanıcı stili:** Kullanıcı stilinde HTML dosyasında herhangi bir stil bildirimi yapılmamışsa ve kullanıcı tarayıcıda stil tanımlamalarında değişiklik yapmışsa, tarayıcıda kullanıcının yapmış olduğu bildiriler etkin olur.
- **Kullanıcı stili / tarayıcı stili:** Tarayıcı stilinde HTML dosyasında herhangi bir stil bildirimi yoktur ve kullanıcı da tarayıcının stil bildirimlerinde değişiklik yapmamıştır, bu dosyanın gösteriminde tarayıcı tarafından varsayılan stil ayarları kullanılır.
- **CSS'DE KALITSALLIK (INHERITANCE)**
- Aynı aile içinde kalıtsal değerlerin büyükten küçüğe geçtiği gibi HTML öğeleri arasında da CSS özellikleri kalıtsal olarak alt etiketleri (çocuk etiket) etkiler. Kalıtsallık CSS kullanımında bir bildirimin tek tek bütün etiketlere uygulanması yerine ata etikete uygulayarak alt (çocuk) etiketlere de otomatik olarak uygulanmasını sağlar. Bir ögenin stilleri alabilmesinin bir yolu da mirastır (kalıttır). Bütün özellikler miras alınmaz, margin (kenar), padding (dolgulama), background (arka plan rengi), width (genişlik) gibi bazı özelliklerde kalıtsallık yoktur.

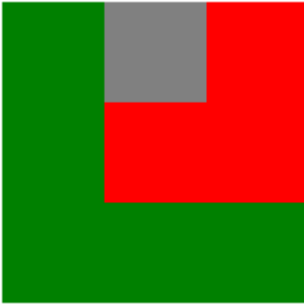
DEĞERLENDİRME SORULARI

1. CSS'teki kutu modeli ile ilgili aşağıdaki ifadelerden hangisi söylenemez?
 - a) Tüm HTML öğeleri kutu olarak kabul edilebilir.
 - b) Kutu modeli, HTML öğelerinin etrafına kenarlık eklenmesine ve elemanlar arasında boşluk tanımlanmasına olanak sağlar.
 - c) Bir ögenin tam boyutunu hesaplamak için, dolgu, kenarlıklar ve kenar boşluklarının da eklenmesi gerekmektedir.
 - d) Bir ögenin genişlik ve yükseklik tanımı ile yalnızca içerik alanının genişliği ve yüksekliği ayarlanmış olur.
 - e) Ok etiketin genişliği veya yüksekliği hesaplanırken kenar boşlukları hesaplanmaz ama dolgular hesaplanır.
2. CSS'teki öncelikler ile ilgili aşağıdaki ifadelerden hangisi söylenemez?
 - a) Öncelikler, oluşturulan bildirimlerde ortaya çıkan çatışmaların nasıl çözüldüğünü belirlemektedir.
 - b) Öncelikler, dilin nasıl çalıştığının temel bir parçasıdır.
 - c) Bir html dosyasında belge düzeyinde stillere göre öncelik baskın olan kısım satır içi stillerdir.
 - d) HTML dosyasında bir stil bildirimi yapılmış olsa bile kullanıcının yapmış olduğu bildirimler etkin olur.
 - e) Bir html dosyasında sayfaya gömülü stiller, CSS dosyasındaki stillere göre baskın olan kısımdır.
3. CSS'teki kalıtsallık ile ilgili aşağıdaki ifadelerden hangisi söylenemez?
 - a) CSS özellikleri kalıtsal olarak alt etiketleri (çocuk etiketleri) etkiler.
 - b) Bir ögenin stilleri alabilmesinin bir yolu da kalıttır.
 - c) Ok <body> etiketi için yapılan bütün bildirimler bu içerikteki bütün etiketler tarafından miras alınır.
 - d) Bir ögenin belirli bir özellik için basamaklı bir değeri yoksa, onu bir ata ögesinden (üst ögeden) miras alabilir.
 - e) CSS'te margin (kenar boşlukları) ve padding (dolgu) değerleri miras alınmaz.

4. Stil tanımı ve sayfa görüntüsü verilen bilgilere göre <div> etiketlerinin html sayfası içerisinde yerleşimi hangisi gibi olabilir?

Stil tanımı	Sayfa görüntüsü
<pre> <style> div.bir { width: 100px; height: 100px; background- color:grey;} div.iki { width: 100px; height: 100px; background-color:red;} div { width: 200px; height: 200px; background-color:green;} </style> </pre>	
<p>a) <div><div class="bir"> </div> <div class="iki"> </div> </div> b) <div></div> <div class="bir"> </div> <div class="iki"> </div> c) <div class="bir"> </div> <div class="iki"> </div><div></div> d) <div class="bir"> <div></div> <div class="iki"> </div> </div> e) <div> <div class="iki"> </div><div class="bir"> </div></div></p>	

5. Stil tanımı ve sayfa görüntüsü verilen bilgilere göre <div> etiketlerinin html sayfası içerisinde yerleşimi hangisi gibi olabilir?

Stil tanımı	Sayfa görüntüsü
<pre> <style> div.bir { width: 50px; height: 50px; background-color:grey;} div.iki { width: 100px; height: 100px; background-color:red; float:right;} div { width: 150px; height: 150px; background-color:green;} </style> </pre>	
<p>a) <div></div><div class="bir"> </div><div class="iki"> </div> b) <div> <div class="iki"> <div class="bir"> </div></div> </div> c) <div class="bir"> </div><div class="iki"> </div><div></div> d) <div class="bir"> <div></div> <div class="iki"> </div> </div> e) <div> <div class="bir"> <div class="iki"> </div></div> </div></p>	

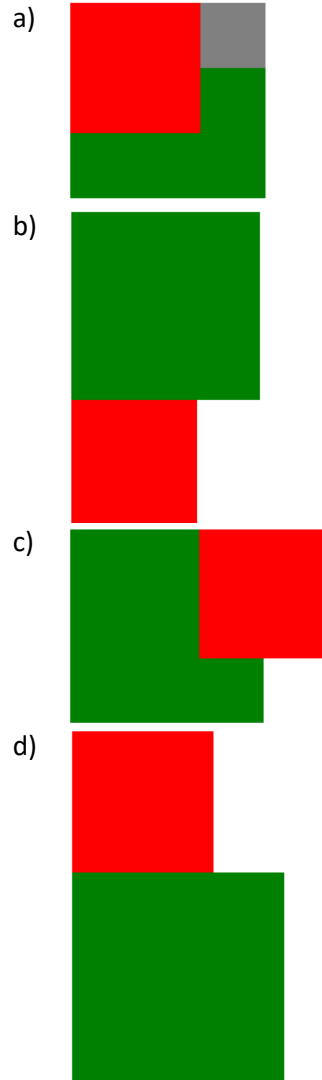
6. Stil tanımı ve sayfa yerleşimi verilen html kodlarının web sayfasında görüntüsü nasıl olabilir?

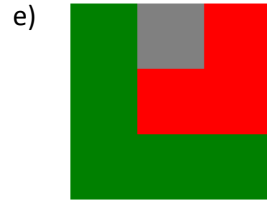
Stil tanımı

```
<style>  
  
div.bir { width: 50px; height: 50px;  
background-color:grey; float:right; }  
  
div.iki { width: 100px; height:  
100px; background-color:red; clear:left}  
  
div { width: 150px; height: 150px;  
background-color:green;}  
</style>
```

Sayfa yerleşimi

```
<div>  
  
<div class="bir"> </div>  
  
<div class="iki"> </div>  
  
</div>
```





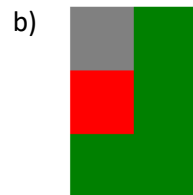
7. Stil tanımı ve sayfa yerleşimi verilen html kodlarının web sayfasında görüntüsü nasıl olabilir?

Stil tanımı

```
<style>  
  
div.bir { width: 100px;  
height: 100px; background-  
color:grey;}  
  
div.iki { width: 50px; height:  
50px; background-color:red;  
position: static; left:100px;}  
  
div { width: 100px; height:  
150px; background-color:green;}  
  
</style>
```

Sayfa yerleşimi

```
<div>  
  
  <div class="bir"> </div>  
  
  <div class="iki"> </div>  
  
</div>
```





8. Sayfa yerleşimi ve sayfa görüntüsü verilen bilgilere göre stil tanımı hangisi gibi olabilir?

Sayfa Yerleşimi

```
<div>
<div class="bir"> </div>
<div class="iki"> </div>
</div>
```

Sayfa görüntüsü



- a) `div.bir { width: 50px; height: 50px; background-color:grey; } div.iki { width: 50px; height: 50px; background-color:red; position: static; left:50px;} div { width: 100px; height: 150px; background-color:green;}`
- b) `div.bir { width: 50px; height: 50px; background-color:grey; } div.iki { width: 50px; height: 50px; background-color:red; position: fixed; left:50px;} div { width: 100px; height: 150px; background-color:green;}`
- c) `div.bir { width: 50px; height: 50px; background-color:grey; } div.iki { width: 50px; height: 50px; background-color:red; position: absolute; left:50px;} div { width: 100px; height: 150px; background-color:green;}`
- d) `div.bir { width: 50px; height: 50px; background-color:grey; } div.iki { width: 50px; height: 50px; background-color:red; position: sticky; left:50px;} div { width: 100px; height: 150px; background-color:green;}`
- e) `div.bir { width: 50px; height: 50px; background-color:grey; } div.iki { width: 50px; height: 50px; background-color:red; position: relative; left:50px;} div { width: 100px; height: 150px; background-color:green;}`

9. Aşağıdaki html sayfasının web sayfası görüntüsü nasıl olabilir?

Stil tanımı

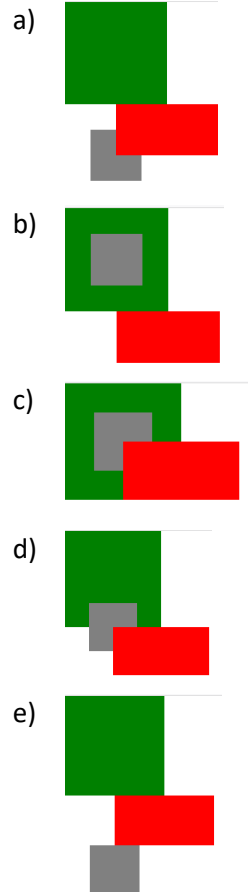
```
<html> <head>
  <meta charset="UTF-8">
  <style>
    body{font-weight: bold; text-decoration: overline;}
    p{font-style: none;}
  </style></head>
<body>
  <p> Atatürk Üniversitesi</p>
  <p style="font-style:italic; font-weight:400;">Açık >Öğretim Fakültesi</p>
</body> </html>
```

- a) Atatürk Üniversitesi
Açık >Öğretim Fakültesi
- b) Atatürk Üniversitesi
Açık >Öğretim Fakültesi
- c) Atatürk Üniversitesi
Açık >Öğretim Fakültesi
- d) Atatürk Üniversitesi
Açık >Öğretim Fakültesi
- e) *Atatürk Üniversitesi*
Açık >Öğretim Fakültesi

10. Aşağıdaki html sayfasının web sayfası görüntüsü nasıl olabilir?

Stil tanımı

```
<html> <head>
  <meta charset="UTF-8">
  <style>
    div.bir { width: 50px; height: 50px; background-color:grey; position:
fixed; left:25px; top:25px;}
    div.iki { width: 50px; height: 50px; background-color:red; position:
relative; top:-50px; left:50px;}
    div { width: 100px; height: 100px; background-color:green;
position:static;}
  </style></head>
<body style="margin:0px; padding:0px;">
  <div></div>
  <div class="bir"> </div>
  <div class="iki"> </div>
</body> </html>
```



Cevap Anahtarı

1.e, 2.d, 3.c, 4.a, 5.b, 6.a, 7.b, 8.e, 9.d, 10.c

YARARLANILAN KAYNAKLAR

Brown, T. B. (2018). CSS master (2nd Edition). VIC: SitePoint Pty. Ltd.

Dean, J. (2019). Web programming with HTML5, CSS, and Javascript. MA: Jones & Bartlett Learning, LLC, an Ascend Learning Company.

Grant, K. J. (2018). CSS in Depth. NY: Manning Publications Co.

Meyer, E. A. (2018). CSS Pocket Reference (5th Edition). CA: O'Reilly Media, Inc.