



CodeCraft

Bootcamp'25

HTML & CSS
Responsive Web Design
3. Gün

CSS Flex

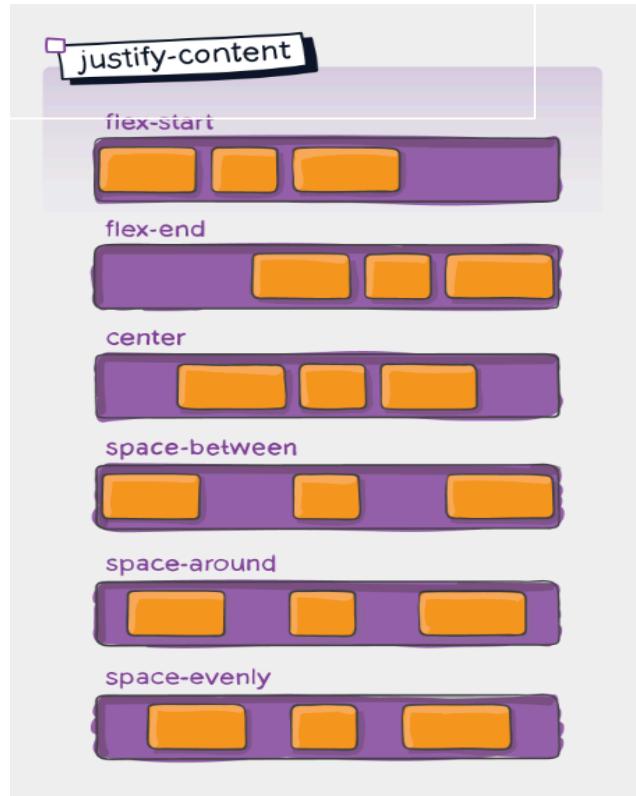
Flexbox, tek boyutlu (yatay veya dikey) hizalama ve dağıtım için geliştirilmiş modern bir CSS düzenleme modelidir.

Esnek ve dinamik yapısı sayesinde, elementleri kolayca hizalamak ve sıralamak için kullanılır.



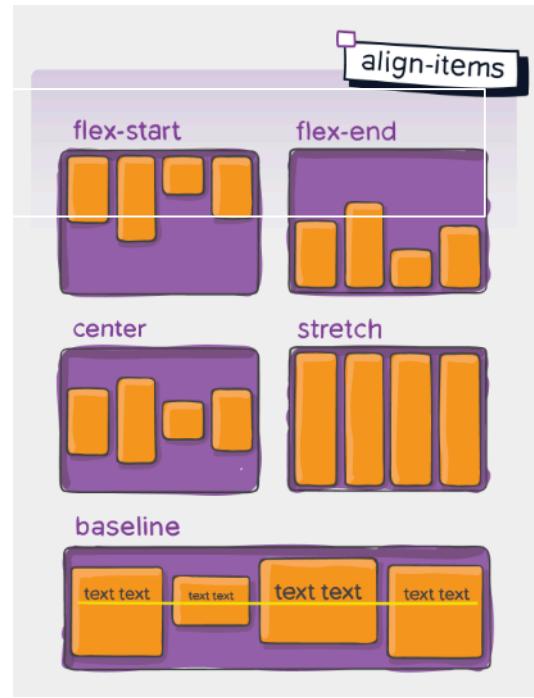
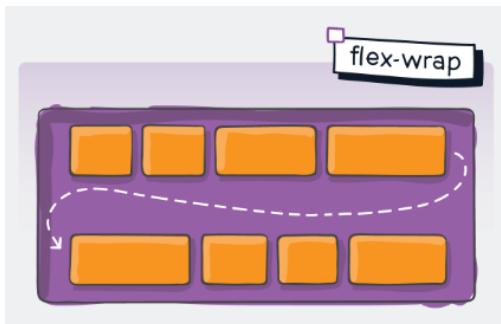
Flexbox Değerleri

- `display: flex;` -> Konteyner'a flex özellik katar.
- `justify-content:` -> Elementleri yatay eksende hizalar:
- `flex-start`: Elementleri sola hizalar.
- `flex-end`: Elementleri sağa hizalar.
- `center`: Elementleri merkeze hizalar.
- `space-between`: İlk ve son öğe kenarlara yaslanır, aradakiler eşit boşluk alır.
- `space-around`: Öğeler arasında ve kenarlarda eşit boşluk bırakır.
- `space-evenly`: Elementleri eşit boşluklarla dağıtır.



Flexbox Değerleri

- **align-items:** -> Öğeleri dikey eksende hizalar:
 - **stretch:** Öğeler konteynerin yüksekliğini doldurur.
 - **flex-start:** Öğeleri üst kenara hizalar.
 - **flex-end:** Öğeleri alt kenara hizalar.
 - **center:** Öğeleri ortalar.
- **flex-wrap:** -> Öğelerin satırlara bölünmesine izin verir (**nowrap**, **wrap**, **wrap-reverse**).
- **align-self:** -> Bireysel öğelerin hizalamasını değiştirmek için kullanılır.
- **order:** -> Öğelerin sırasını değiştirir.

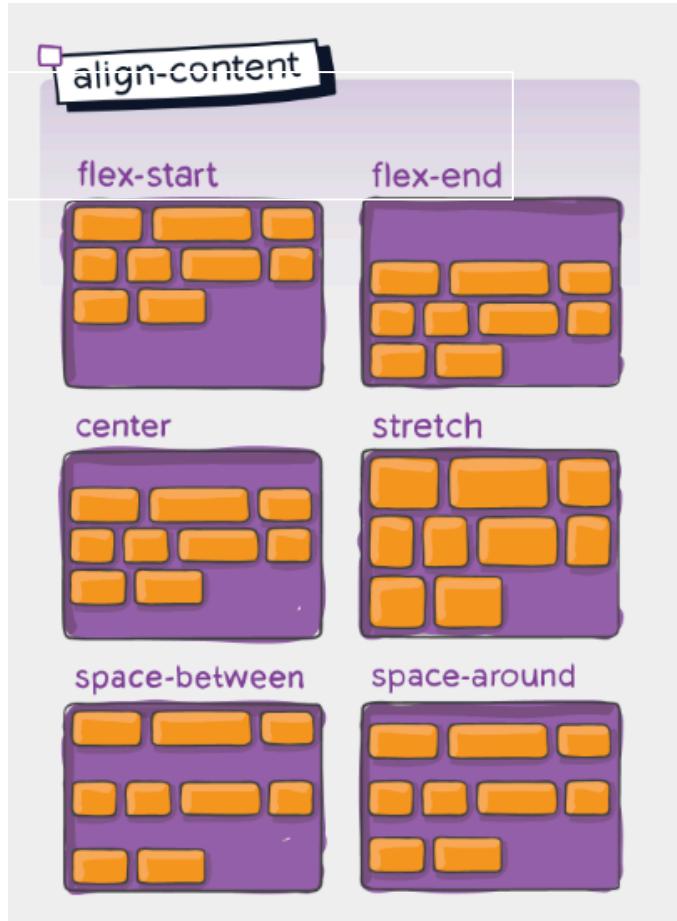


Flexbox Değerleri

align-content: -> Çok satırlı flex konteynerlerde satırların hizalanmasını belirler:

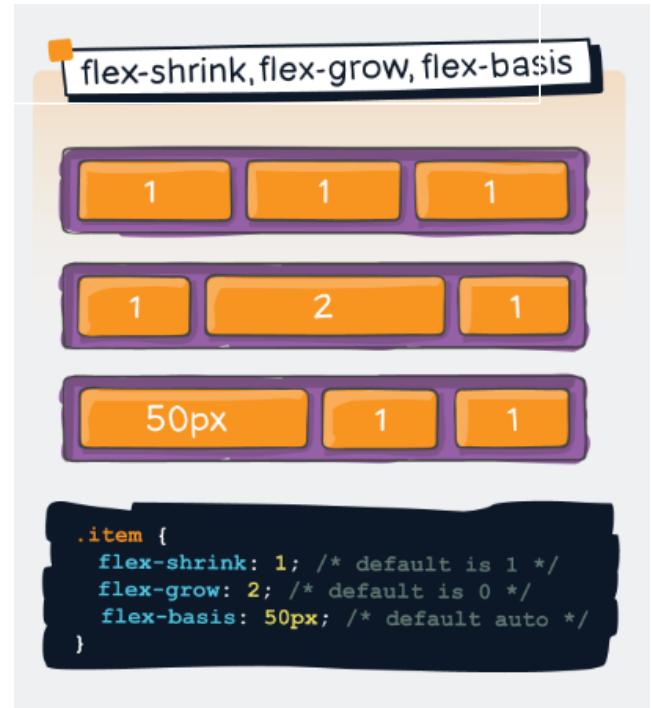
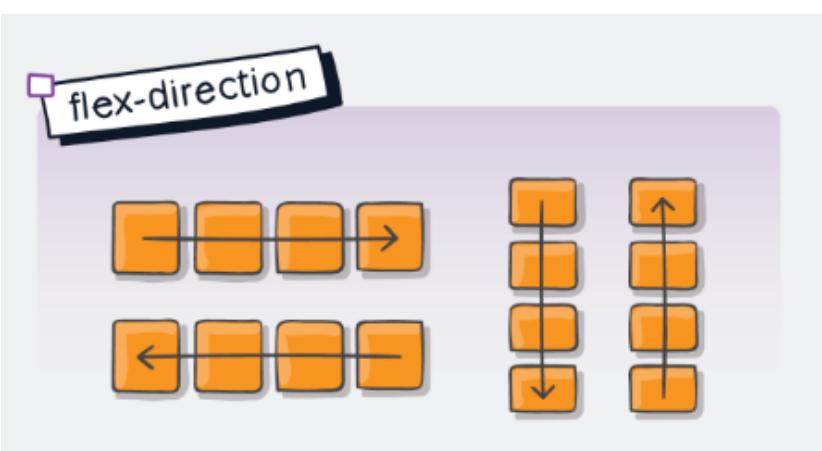
- **flex-start:** Satırları konteynerin üstüne hizalar.
- **flex-end:** Satırları konteynerin altına hizalar.
- **center:** Satırları konteynerin ortasına hizalar.
- **space-between:** İlk ve son satırı kenarlara yaslar, aradakileri eşit boşluklarla ayırrır.
- **space-around:** Satırlar arasında ve kenarlarda eşit boşluk bırakır.
- **stretch:** Satırları mümkün olan en fazla alanı kaplayacak şekilde uzatır.

Alıştırma -> <https://flexboxfroggy.com/>



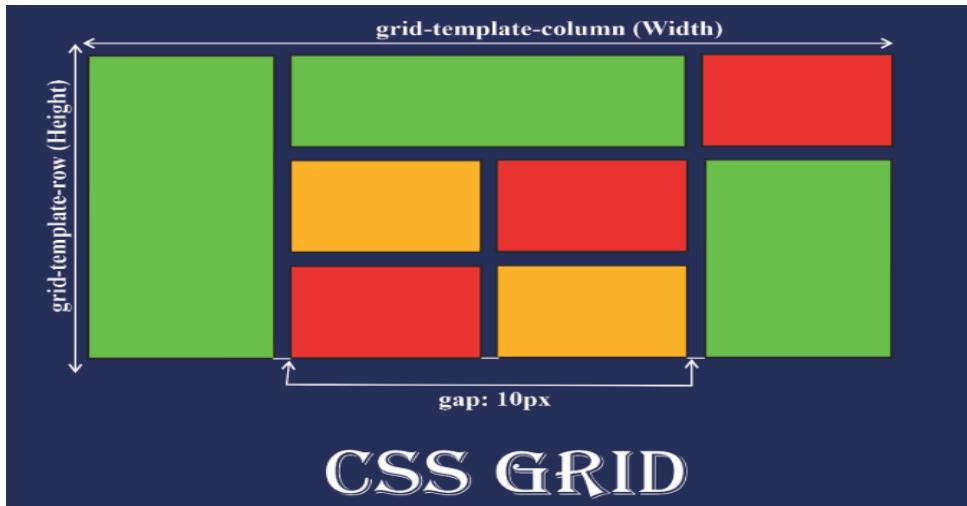
Flexbox Değerleri

- **flex-grow:** -> Öğelerin genişliğini belirler.
- **flex-shrink:** -> Öğelerin küçülmesini kontrol eder.
- **flex-basis:** -> Ögenin başlangıç genişliğini belirler
- **flex-direction:** -> Elementlerin yönünü belirler (**row**, **row-reverse**, **column**, **column-reverse**).



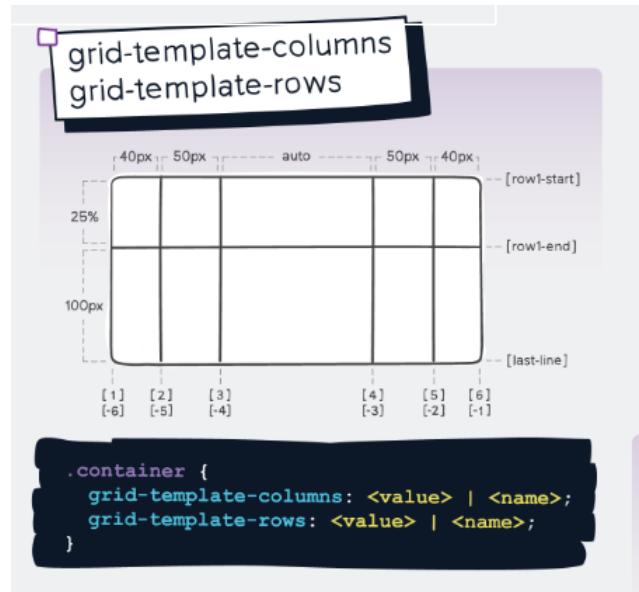
CSS Grid

CSS Grid, iki boyutlu (hem satır hem sütun) düzenlemeye yarayan bir layout sistemidir.



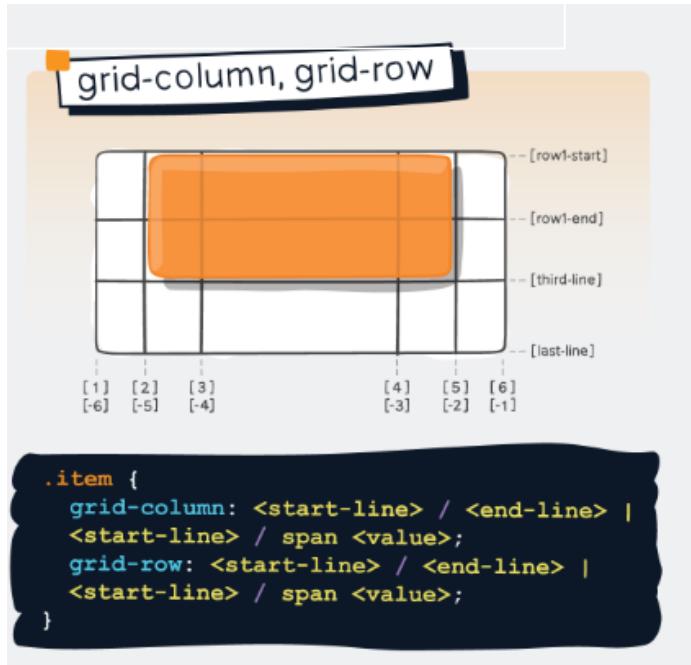
CSS Grid Değerleri

- `display: grid;` -> Konteyneri bir grid konteyner yapar.
- `grid-template-columns:` -> Sütun sayısını ve genişliğini belirler.
- `grid-template-rows:` -> Satır sayısını ve yüksekliğini belirler.



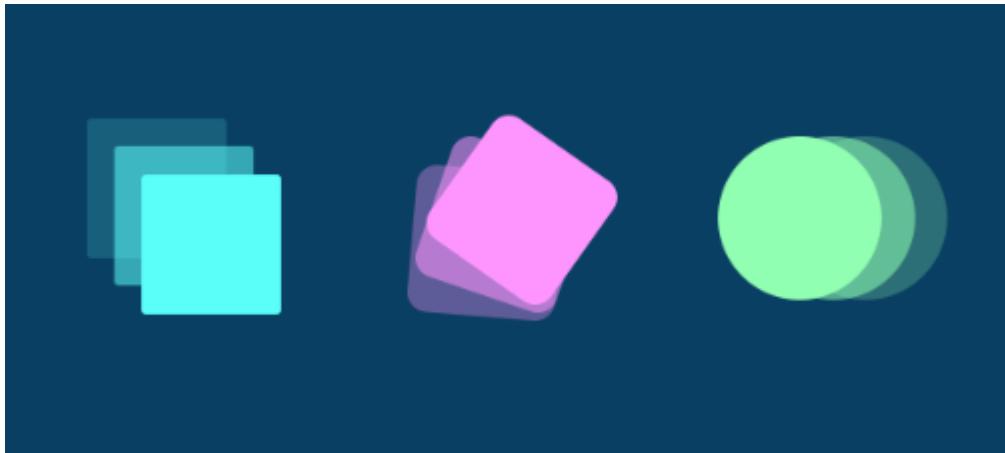
CSS Grid Değerleri

- **gap:** -> Satır ve sütunlar arasındaki boşluğu ayarlar.
- **grid-auto-flow:** -> Öğelerin otomatik akış yönünü belirler.
- **grid-column:** -> Bir öğenin kaç sütun kaplayacağını belirler.
- **grid-row:** -> Bir öğenin kaç satır kaplayacağını belirler.



CSS Animations

Seçili elemente animasyonlar eklenmesini sağlar.



CSS Animations

- **animation-name**: Kullanılacak animasyonun adını belirler.
- **animation-duration**: Animasyonun süresini belirler (ör: **2s**, **500ms**).
- **animation-timing-function**: Animasyonun hız eğrisini belirler (**ease**, **linear**, **ease-in**, **ease-out**, **ease-in-out**, **cubic-bezier()**).
- **animation-delay**: Animasyonun başlamadan önce bekleme süresini belirler.

CSS Animations

- **animation-iteration-count**: Animasyonun tekrar sayısını belirler (1, infinite).
- **animation-direction**: Animasyonun yönünü belirler (normal, reverse, alternate, alternate-reverse).
- **animation-fill-mode**: Animasyon tamamlandıktan sonra öğenin nasıl görüneceğini belirler (none, forwards, backwards, both).
- **animation-play-state**: Animasyonu durdurmak veya devam ettirmek için kullanılır (running, paused).

CSS Animations

@keyframes

- CSS'te animasyonlar oluşturmak için kullanılan bir kuraldır.
- Bu kural, bir ögenin belirli zaman dilimlerinde nasıl değişeceğini tanımlamak için kullanılır.

CSS Animations

`@keyframes hareket {}` → "hareket" adında bir animasyon tanımlar.

`from` ve `to` → Animasyonun başlangıç (`from`) ve bitiş (`to`) durumlarını belirler.

```
.box { animation: hareket 2s linear infinite; }
```

- **hareket** → Kullanılacak animasyonun adı.
- **2s** → Animasyonun süresi (2 saniye).
- **linear** → Animasyonun hız eğrisi (düzgün hızda).
- **infinite** → Animasyonun sürekli tekrarlanmasını sağlar.

```
@keyframes hareket {  
    from {  
        transform: translateX(0);  
    }  
    to {  
        transform: translateX(200px);  
    }  
}  
  
.box {  
    width: 100px;  
    height: 100px;  
    background-color: red;  
    animation: hareket 2s linear infinite;  
}
```

CSS Animations

@keyframes Çeşitleri:

1. **Linear Animation:** Animasyon baştan sona sabit hızda çalışır.
2. **Ease Animation:** Hareketin başı ve sonu yavaş, ortası hızlı olur.
3. **Steps Animation:** Animasyonu belirli adımlarla oynatır.
4. **Bounce Animation:** Öğenin zıplayan bir efekt almasını sağlar.
5. **Fade Animation:** Öğeyi şeffaflıktan görünür hale getirir.
6. **Rotate Animation:** Öğeyi belirli bir açıda döndürür.

<https://acchou.github.io/html-css-cheat-sheet/animation.html#animation>

<https://animate.style/>

CSS Transform

Transform

- Bir HTML öğesinin döndürülmesi (rotate), ölçeklenmesi (scale), taşınması (translate) ve eğilmesi (skew) gibi dönüşümleri gerçekleştirmek için kullanılan bir CSS özelliğidir.
- Bu özellik, ögenin düzenini (layout) değiştirmeden görsel olarak farklı şekillerde görüntülenmesini sağlar.

CSS Transform

translate(x, y) Konum değiştirmemizi sağlar

Belirtilen x ve y eksenlerinde öğeyi taşıır.

- **Pozitif değerler** öğeyi sağa ve aşağı kaydırır.
- **Negatif değerler** öğeyi sola ve yukarı kaydırır.

```
.box {  
    transform: scale(1.5, 2); /* Genişlik 1.5 kat, yükseklik 2 kat */  
}
```

✓ Alternatif:

- **translateX(50px)** → **Sadece X ekseninde hareket ettirir.**
- **translateY(20px)** → **Sadece Y ekseninde hareket ettirir.**

CSS Transform

scale(x, y) - Boyut Değiştirme

Öğenin genişlik ve yüksekliğini ölçeklendirir.

```
.box {  
    transform: scale(1.5, 2); /* Genişlik 1.5 kat, yükseklik 2 kat */  
}
```

✓ Alternatif:

- `scaleX(1.5)` → Sadece yatay eksende büyütür/küçültür.
- `scaleY(2)` → Sadece dikey eksende büyütür/küçültür.

CSS Transform

rotate(angle) - Döndürme

Öğeyi belirli bir açıyla saat yönünde veya tersine döndürür.

```
.box {  
    transform: rotate(45deg); /* Saat yönünde 45 derece döndür */  
}
```

✓ Alternatif:

- `rotate(-30deg)` → Saat yönünün tersinde 30 derece döndürür.

CSS Transform

skew(x, y) - Eğme

Öğeyi X veya Y eksenin boyunca eğerek (distortion) görüntüsünü değiştirir.

```
.box {  
    transform: skew(30deg, 15deg); /* X ekseninde 30°, Y ekseninde 15° eğ */  
}
```

✓ Alternatif:

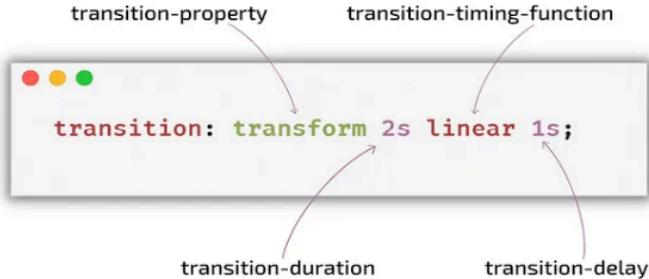
- `skewX(30deg)` → **Sadece X ekseninde eğme.**
- `skewY(15deg)` → **Sadece Y ekseninde eğme.**

CSS Transition

Transition bir CSS özelliğinin belirli bir süre içinde, yumuşak bir geçişle (animasyon olmadan) değişmesini sağlar.

Bu sayede öğeler, aniden değil, akıcı bir şekilde değişir.

CSS Transition



Parametre	Açıklama	Örnek
property	Hangi CSS özelliğinin geçiş yapacağını belirler	width, opacity, background-color
duration	Geçiş süresi	0.5s, 2s, 300ms
timing-function	Geçiş eğrisi (hızlanma/yavaşlama)	ease, linear, ease-in, ease-out, ease-in-out
delay	Geçişin ne kadar gecikeceğini belirler	0s, 1s

CSS Media Queries

Media Queries, ekran boyutuna göre farklı stiller uygulanmasına olanak tanır.

Media Queries Attributeleri ve Kullanımı:

- `min-width`: Belirtilen genişlikten büyük ekranlar için stil uygular.
- `max-width`: Belirtilen genişlikten küçük ekranlar için stil uygular.
- `min-height`: Belirtilen yükseklikten büyük ekranlar için stil uygular.
- `max-height`: Belirtilen yükseklikten küçük ekranlar için stil uygular.
- `orientation`: Cihazın yönünü belirler (`portrait`, `landscape`).
- `aspect-ratio`: Ekran oranına göre stiller uygular.
- `hover`: Cihazın hover (fare imleci) destegine göre stiller uygular (`none`, `hover`).

```
@media (min-width: 300px) and (max-width: 750px) {  
    ...  
}
```