

Flexbox ile Modern Düzen

Ana ve çapraz eksen ■

Esneklik: / /

■

■

■

■

Prof. Dr. Özal YILDIRIM

Fırat Üniversitesi — Teknoloji Fakültesi

Yazılım Mühendisliği Bölümü

Geçmişte Düzen Kurmak Neden Zordu?

Geliştiriciler, **aslında bu iş için tasarlanmamış** `float` ve `position` özelliklerini kullanarak “hileli” çözümler üretmek zorunda kalıyordu.

🕒 Eski Dünya (2000–2015)

- ▶ `float: left` ile sütun kurma
- ▶ `clearfix` hilesi
- ▶ Dikey ortalama için `table-cell` numarası
- ▶ Eşit yükseklikli sütunlar için JavaScript
- ▶ Her şey **kırılgan ve yönetilmesi zor**

★ Modern Çözüm

Bu sorunları özel olarak çözmek için **iki modül** geliştirildi: **Flexbox** — **tek boyutlu** düzen (bu hafta)

Grid — **iki boyutlu** düzen (gelecek hafta)

Bu hafta sonunda: **Menüyü yatay dizecek**, kartları yan yana sıralayacak ve her şeyi ortalayabileceksiniz.

- ▶ Flex kapsayıcı ve flex öge kavramlarını ayırt edebilecek,
- ▶ Ana eksen ve çapraz eksen kavramını `flex-direction` ile ilişkilendirebilecek,
- ▶ `justify-content` ve `align-items` ile hizalama yapabilecek,
- ▶ `flex-wrap` ve `gap` ile sarılan esnek ızgaralar kurabilecek,
- ▶ `flex-grow`, `flex-shrink`, `flex-basis` üçlüsünü açıklayabilecek,
- ▶ `order` ve `align-self` ile tekil öğeleri yönetebilecek,
- ▶ Yatay menü, kart ızgarası ve yapışkan footer düzenlerini kurabilecek,
- ▶ Yapay zekaya Flexbox çözümü ürettirip doğrulayabileceksiniz.

Bu Haftanın İçeriği

- 1 Flexbox Temelleri
- 2 Öğe Özellikleri
- 3 Uygulama: BookVerse
- 4 Yapay Zeka ile Flexbox
- 5 Özet ve Değerlendirme



Flexbox Temelleri

BÖLÜM 1

Tanım

Flexbox (Esnek Kutu Modeli), elementleri bir kapsayıcı içinde **tek bir boyut** (satır veya sütun) boyunca hizalamak, sıralamak ve aralarındaki boşluğu dağıtmak için tasarlanmıştır.

Flex Container

`display: flex` uyguladığımız **ana, sarmalayıcı** elementtir.

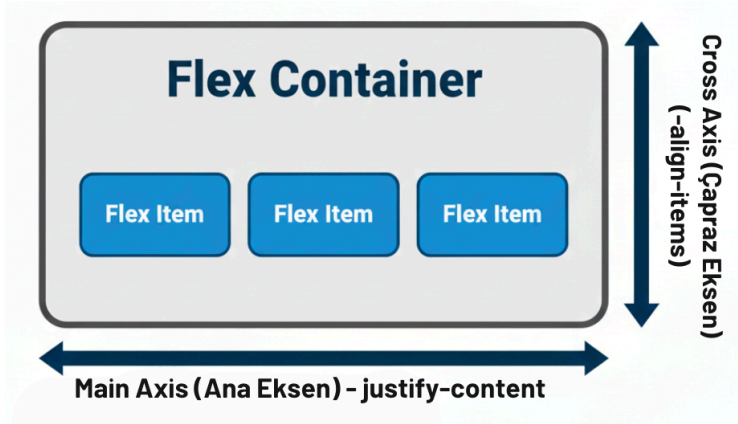
```
.kapsayici { display: flex; }
```

Flex Items

Kapsayıcının **doğrudan alt elemanlarıdır**. Torunlar etkilenmez!

```
<div class="kapsayici">  
  <div>Öge 1</div>  <!-- flex item -->  
  <div>Öge 2</div>  <!-- flex item -->  
</div>
```

Anahtar kelime: TEK BOYUT. Bir menüdeki elemanları yan yana dizmek veya kartları eşit aralıklarla sıralamak için mükemmeldir.



Kritik nokta: Eksenler `flex-direction` değerine göre **yer değiştirir**. Bu, Flexbox'ta en çok kafa karıştıran konudur.

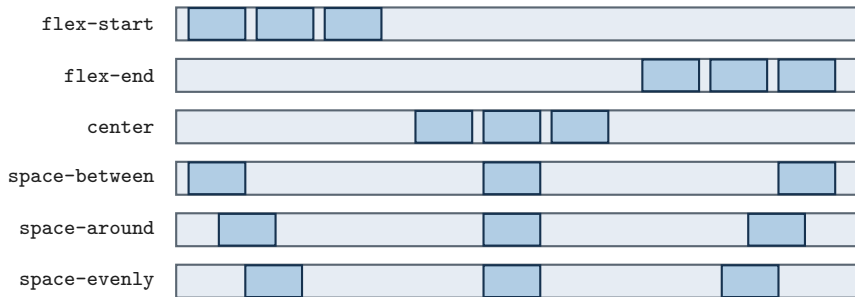
flex-direction: Eksenleri Belirler

```
.kapsayici { display: flex; flex-direction: row; } /* varsayılan */
```

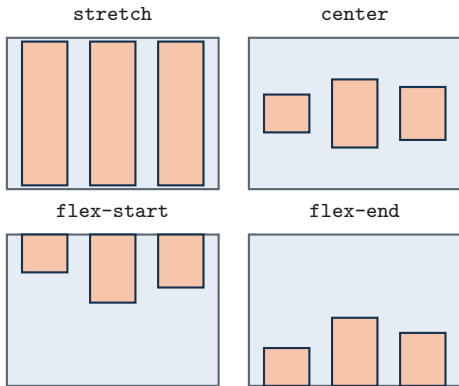
flex-direction	Ana eksen	Çapraz eksen	Diziliş
row (varsayılan)	Yatay (→)	Dikey (↓)	Soldan sağa
row-reverse	Yatay (←)	Dikey (↓)	Sağdan sola
column	Dikey (↓)	Yatay (→)	Yukarıdan aşağı
column-reverse	Dikey (↑)	Yatay (→)	Aşağıdan yukarı

⚠ Dikkat

Altın kural: justify-content her zaman ana eksenini, align-items her zaman çapraz eksenini hizalar. flex-direction: column yazdığınızda justify-content artık **dikeyde** çalışır! Bu karıştırılmazsa Flexbox kolaylaşır.



space-between: uçlara yaslı, **aralarda eşit** ▪ **space-around**: her öğenin iki yanında eşit (**uçlarda yarım**) ▪
space-evenly: **tüm** boşluklar eşit



Değer

Etki

	Varsayılan. Öğeler kapsayıcı yüksekliğine uzar
stretch	
flex-start	Çapraz eksenin başına yaslanır
flex-end	Sonuna yaslanır
center	Ortalanır — en sık kullanılan
baseline	Metin taban çizgilerine göre hizalanır

İpucu

Efsanevi ortalama:

```
.orta {  
  display: flex;  
  justify-content: center;  
  align-items: center;  
}
```

Prof. Dr. Özalp YILDIRIM

gap: Boşluk Vermenin Doğru Yolu

✗ Eski yöntem

```
.oge { margin-right: 20px; }  
.oge:last-child { margin-right: 0; }
```

Son ögeyi ayrıca ele almak gerekir. Sarılma (wrap) durumunda dikey boşluk ayrı yazılır.

✓ Modern yöntem

```
.kapsayici {  
  display: flex;  
  gap: 20px;           /* her yon */  
  gap: 12px 24px;      /* satir sutun */  
  row-gap: 12px;  
  column-gap: 24px;  
}
```

Sadece aralara boşluk koyar; uçlarda boşluk bırakmaz.

💡 İpucu

gap, Flexbox ve Grid'de **margin çökmesi yaşamaz** ve her zaman öngörülebilirdir. Boşluk vermek için **önce gap'i düşünün**, margin'i sadece tekil istisnalar için kullanın.

flex-wrap: Satıra Sığmayanlar

```
.kapsayici {  
  display: flex;  
  flex-wrap: nowrap;    /* VARSAYILAN: tek satırda sığdır */  
  flex-wrap: wrap;      /* sığmayanlar alt satıra geçsin */  
  flex-wrap: wrap-reverse;  
}  
  
/* Kisa yazim: direction + wrap */  
.kapsayici { flex-flow: row wrap; }
```

✂ nowrap Tuzağı

Varsayılan **nowrap**'tır. Öğeler sığmaz ise **ezilerek küçülür** — alt satıra geçmez. Kart ızgaralarında **mutlaka wrap** yazın.

✂ Duyarlı Kart Izgarası

```
.kartlar {  
  display: flex;  
  flex-wrap: wrap;  
  gap: 1.5rem;  
}  
.kart {  
  flex: 1 1 260px;    /* min 260px, esner */  
}
```

Media query **olmadan** duyarlı ızgara!

Prof. Dr. Özal YILDIRIM

align-content: Çok Satırlı Hizalama

`align-items` **her satırın içindeki** öğeleri hizalar. `align-content` ise **satırların kendisini** hizalar — sadece `flex-wrap: wrap` varken çalışır.

```
.kapsayici {  
  display: flex;  
  flex-wrap: wrap;  
  height: 500px;  
  align-content: center; /* satirlarin tamamini dikeyde ortalama */  
  /* flex-start | flex-end | center | space-between | space-around | stretch */  
}
```

⚠ Dikkat

Karıştırmayın:

`align-items` → tek satır içi hizalama

`align-content` → satırlar arası hizalama

💡 İpucu

Tek satırlık bir flex kapsayıcıda `align-content` **hiçbir şey yapmaz**. Etkisiz kaldığında önce `flex-wrap: wrap` olup olmadığını kontrol edin.

Karşılaştırma: Eski Yöntem vs Flexbox

Aynı iş: **Üç eşit sütun, aralarında 20px boşluk.**

✗ float ile (2010)

```
.satir { overflow: hidden; } /* clearfix */
.sutun {
  float: left;
  width: calc(33.333% - 13.34px);
  margin-right: 20px;
}
.sutun:nth-child(3n) { margin-right: 0; }
/* Esit yukseklik icin JavaScript gerekir */
/* Dikey ortalama icin table-cell hilesi */
```

✓ Flexbox ile (bugün)

```
.satir { display: flex; gap: 20px; }
.satir > * { flex: 1; }
```

- ▶ Hesap yok
- ▶ Clearfix yok
- ▶ Eşit yükseklik **bedava**
- ▶ Dikey ortalama **bir satır**

9 satır + JavaScript yerine **2 satır CSS**. Eski projelerde *float* tabanlı kod görürseniz şaşırmayın — ama yenisini **asla** böyle yazmayın.

Duyarlı Menü Hazırlığı

```
1  /* Genis ekran: yatay menu */
2  .ust-bar {
3      display: flex;
4      align-items: center;
5      justify-content: space-between;
6      gap: var(--bosluk-m);
7      flex-wrap: wrap;                /* dar ekranda alt satira gecsin */
8  }
9
10 .ust-bar nav ul {
11     display: flex;
12     gap: var(--bosluk-s);
13     list-style: none;
14     margin: 0; padding: 0;
15     flex-wrap: wrap;
16 }
17
18 /* Dar ekranda dikey menu - Hafta 9'da detaylandiracagiz */
19 @media (max-width: 640px) {
20     .ust-bar { flex-direction: column; align-items: stretch; }
21     .ust-bar nav ul { flex-direction: column; }
22 }
```



Öğe Özellikleri

BÖLÜM 2

flex: grow / shrink / basis

```
.oge { flex: 1 1 200px; }  
/*      grow shrink basis */
```

Özellik	Varsayılan	Anlamı
flex-grow	0	Fazla boşluk varsa ne kadar büyüsün? Oran belirtir
flex-shrink	1	Yer yetmezse ne kadar küçülsün? 0 = küçülme
flex-basis	auto	Başlangıç boyutu (width 'in önüne geçer)

```
.oge { flex: 1; }           /* = 1 1 0%    - tum ogeler ESIT genislik */  
.oge { flex: auto; }       /* = 1 1 auto  - icerige gore, sonra esner */  
.oge { flex: none; }       /* = 0 0 auto  - hic esnemesin */  
.oge { flex: 2; }          /* digerlerinin IKI KATI paya sahip */
```

En çok kullanılan iki kalıp: flex: 1 (eşit sütunlar) ve flex: 1 1 260px (duyarlı kartlar).

Tekil Öğe Kontrolü

```
1 /* Sadece bu ogeyi capraz ekseninde farkli hizala */
2 .ozel { align-self: flex-end; }      /* auto | flex-start | center | stretch */
3
4 /* Gorsel siralamayi degistir (varsayilan 0) */
5 .ilk-gorunsun { order: -1; }
6 .son-gorunsun { order: 99; }
7
8 /* Sag tarafa itmenin klasik hilesi */
9 .sag-blok { margin-left: auto; }
```

↔ margin: auto Hilesi

Flexbox'ta `margin-left: auto`, o öğeyi ve sonrasını **sağa iter**. `justify-content` kullanmadan tek bir öğeyi ayırmak için idealdir.

⚠ order Uyarısı

`order` sadece **görsel sırayı** değiştirir. DOM sırası aynı kalır — **klavye ile Tab sırası** ve ekran okuyucu **eski sırayı** izler. Görsel ve mantıksal sıra ayrışırsa **erişilebilirlik hatası** oluşur.

Uygulama: BookVerse

BÖLÜM 3

Yatay Menü Kurma

```
1 header {
2   background-color: #f8f9fa;
3   padding: 20px;
4   border-bottom: 1px solid #dddddd;
5
6   display: flex;                /* Flexbox'i aktive ediyoruz */
7   justify-content: space-between; /* Ogeleri iki uca yasla */
8   align-items: center;          /* Dikeyde ortalama */
9 }
10
11 header nav ul {
12   list-style: none;             /* Liste noktalarini kaldır */
13   margin: 0;
14   padding: 0;
15   display: flex;               /* Menu de kendi icinde flex */
16   gap: 20px;                  /* Elemanlar arasi bosluk */
17 }
```

1 `<header>`'a `display: flex` verdik — logo ve menü artık **yan yana**.

2 `space-between` ile ikisini **iki uca yasladık**.

3 `align-items: center` ile dikeyde **ortaladık**.

4 `<ul>`'yi de flex yaparak `<li>`'leri yan yana dizdik; `gap` ile ayırdık.



[Ana Sayfa](#) [Kitaplar](#) [Hakkımızda](#) [İletişim](#)

BookVerse'e Hoş Geldiniz

Burasi, kitaplar hakkında yorumlarinizi paylasabileceginiz, yeni yazarlar kefedebileceginiz ve kitap tutkunlariyla bulusabileceginiz bir platformdur.

Haftanın Çok Satanları

Bultenimize Kayıt Olun

```
1 .kitap-listesi {
2   display: flex;
3   flex-wrap: wrap;
4   gap: var(--bosluk-m);
5   list-style: none;
6   padding: 0;
7 }
8
9 .kitap-kart {
10  flex: 1 1 240px;      /* min 240px, esner */
11  max-width: 320px;
12
13  display: flex;         /* KART DA flex */
14  flex-direction: column;
15
16  padding: var(--bosluk-s);
17  border: 1px solid var(--renk-cizgi);
18  border-radius: var(--yuvarlak);
19 }
20
21 .kitap-kart p { flex-grow: 1; } /* metin ğbosluu doldursun */
22
23 .kitap-kart .buton {
24   margin-top: auto;      /* butonu EN ALTA it */
25 }
```

İç İçe Flexbox

Kartın **kendisi de** bir flex kapsayıcısı. **column** yönünde dizilir. **margin-top:**

auto butonu **kartın en altına** iter — böylece farklı uzunluktaki metinlere rağmen tüm butonlar **aynı hizada** durur.

Eşit Yükseklik

align-items: stretch varsayılan olduğu için aynı satırdaki tüm kartlar **otomatik olarak eşit yükseklikte** olur. Eskiden bu, JavaScript gerektiriyordu.

Yapışkan Footer (Sticky Footer)

Kısa içerikli sayfalarda footer'ın **ekranın ortasında** asılı kalmasını engeller.

```
1 body {
2   min-height: 100vh;
3   display: flex;
4   flex-direction: column;
5 }
6
7 main {
8   flex: 1;           /* Kalan TUM ğbosluu kapla */
9 }
10
11 footer {
12   /* Otomatik olarak en altta kalir */
13   background: var(--renk-ana);
14   color: #fff;
15   padding: var(--bosluk-m);
16 }
```

Altı satırlık çözüm. Yıllarca “sticky footer hack” diye aranılan şey, Flexbox ile **apaçık** hale geldi.

Tam ortalama

```
.orta { display: flex;
        justify-content: center;
        align-items: center;
        min-height: 100vh; }
```

İkon + metin hizalama

```
.etiket { display: inline-flex;
          align-items: center;
          gap: 0.4em; }
```

Form satırı (input + buton)

```
.arama { display: flex; gap: 0.5rem; }
.arama input { flex: 1; }
.arama button { flex: none; }
```

Kenar çubuğu + içerik

```
.duzen { display: flex; gap: 2rem; }
.yan    { flex: 0 0 260px; } /* sabit */
.ana    { flex: 1; }        /* kalan */
```

Eşit üç sütun

```
.satir { display: flex; gap: 1rem; }
.satir > * { flex: 1; }
```

Sağa itilmiş son öğe

```
.ustbar { display: flex; align-items: center; }
.ustbar .giris { margin-left: auto; }
```

Bu altı kalıp, gerçek projelerde karşılaşacağınız düzen ihtiyacının **%80'ini** karşılar.

🔍 Nasıl Açılır?

- 1 `display: flex` olan bir elementi seçin
- 2 Elements panelinde yanında **flex** rozeti belirir
- 3 Rozete tıklayın — kapsayıcının üzerine **çizgiler ve boşluklar** çizilir
- 4 Styles panelinde `justify-content` yanındaki ikonla değerleri **tek tıkla** deneyin

💡 Ne Öğrenirsiniz?

- ▶ Ana eksenin **hangi yönde** olduğu
- ▶ **gap** boşluklarının nerede olduğu
- ▶ Hangi öğenin **ne kadar büyüdüğü**
- ▶ Sarılma (**wrap**) satır sınırları

Önerim: Bu haftaki tüm alıştırmaları denetçi açıkken yapın.

Ekstra kaynak: flexboxfroggy.com — 24 seviyelik Flexbox oyunu. Bir saatte tüm özellikleri pekiştirir.

Sık Karşılaşılan Flexbox Tuzakları

Belirti	Sebebe ve Çözüm
Öğeler alt satıra geçmiyor, eziliyor	<code>flex-wrap: nowrap</code> varsayılan. wrap ekleyin.
Uzun metin kapsayıcıdan taşıyor	Flex öğelerinin <code>min-width</code> değeri <code>auto</code> 'dur. <code>min-width: 0</code> veya <code>overflow: hidden</code> ekleyin.
<code>align-items</code> çalışmıyor	Yanlış eksen. <code>flex-direction: column</code> ise <code>justify-content</code> deneyin.
<code>justify-content</code> etkisiz	Öğeler zaten tüm alanı dolduruyor (<code>flex: 1</code> verilmiş). flex'i kaldırın.
Görsel ezilip bozuluyor	<code>flex-shrink</code> varsayılan 1. <code>flex-shrink: 0</code> verin.
Alt eleman flex öğesi olmuyor	Sadece doğrudan çocuklar flex öğesidir. Torunlar için ayrı kapsayıcı gerekir.
<code>gap</code> çalışmıyor	Çok eski tarayıcı. caniuse.com kontrol edin (bugün %97+ destek).

En sıkısı: `min-width: auto` tuzağı. Bir kart içindeki uzun başlık kartı genişletiyorsa `min-width: 0` yazın.

⚠ Görsel-DOM Ayrışması

`order`, `row-reverse` ve `column-reverse` **sadece görsel** sırayı değiştirir. **Klavye Tab** sırası ve **ekran okuyucu** hâlâ **HTML sırasını** izler. Kullanıcı soldaki butona bakarken Tab sağdaki butona giderse **kaybolur**.

✓ Doğru Yaklaşım

- 1 HTML'i **mantıksal okuma sırasına** göre yazın
- 2 Görsel düzeni CSS ile kurun
- 3 `order` kullanacaksanız, **Tab ile test edin**
- 4 `list-style: none` verdiğiniz `<ul>`'ye Safari için `role = "list"` ekleyin

```
<nav aria-label="Ana menu">
  <ul role="list">      <!-- Safari'de liste ğsemantii korunur -->
    <li><a href="/">Ana Sayfa</a></li>
  </ul>
</nav>
```

Uygulama: BookVerse Footer

```
<footer class="site-footer">
  <div class="footer-ust">
    <div class="footer-blok">
      <h3>BookVerse</h3>
      <p>Kitap tutkunlarının buluşma noktası.</p>
    </div>
    <div class="footer-blok">
      <h4>Kesfet</h4>
      <ul role="list">
        <li><a href="#">Kitaplar</a></li>
        <li><a href="#">Yazarlar</a></li>
      </ul>
    </div>
    <div class="footer-blok">
      <h4>Kurumsal</h4>
      <ul role="list">
        <li><a href="#">Hakkımızda</a></li>
        <li><a href="#">İletişim</a></li>
      </ul>
    </div>
  </div>
  <div class="footer-alt">
    <p>2026 BookVerse</p>
    <p>Tüm hakları saklıdır.</p>
  </div>
</footer>
```

```
.site-footer {
  background: var(--renk-ana);
  color: #fff;
  padding: var(--bosluk-l) var(--bosluk-m);
}

.footer-ust {
  display: flex;
  flex-wrap: wrap;
  gap: var(--bosluk-l);
  justify-content: space-between;
}

.footer-blok { flex: 1 1 220px; }

.footer-blok ul {
  list-style: none;
  padding: 0;
  display: flex;
  flex-direction: column;
  gap: 0.4rem;
}

.footer-alt {
  display: flex;
  justify-content: space-between;
  flex-wrap: wrap;
  gap: var(--bosluk-s);
  margin-top: var(--bosluk-l);
  padding-top: var(--bosluk-s);
}
```

Flexbox Hata Ayıklama Kontrol Listesi

Düzeniniz beklendiği gibi çalışmıyorsa **sırayla** kontrol edin:

☰ 1–4: Yapı

- 1 `display: flex` **doğru elemente** mi verilmiş? (Kapsayıcıya, öğeye değil)
- 2 Stil verdiğiniz öğeler **doğrudan çocuk** mu?
- 3 `flex-direction` ne? Ana eksen hangi yönde?
- 4 Kapsayıcının bir **yüksekliği** var mı? (`align-items` için gerekli olabilir)

☰ 5–8: Davranış

- 5 `flex-wrap` gerekli mi? Ekli mi?
- 6 Öğelerde `flex` tanımlı mı? (`flex: 1` varsa `justify-content` etkisiz kalır)
- 7 Taşan metin için `min-width: 0` gerekiyor mu?
- 8 DevTools **flex rozetine** tıklayıp ölçün — tahmin etmeyin

💡 İpucu

Hızlı teşhis: Kapsayıcıya geçici olarak `outline: 2px solid red;` ve öğelere `outline: 1px solid blue;` verin. Kim kapsayıcı, kim öğe — bir bakışta görünür.

Flexbox ile Form Düzeni

```
1  /* Dikey form akisi */
2  .form-alan {
3    display: flex;
4    flex-direction: column;
5    gap: 0.35rem;
6    margin-bottom: var(--bosluk-s);
7  }
8
9  /* Yan yana iki alan (ad + soyad) */
10 .form-satir {
11   display: flex;
12   gap: var(--bosluk-s);
13   flex-wrap: wrap;
14 }
15 .form-satir > * { flex: 1 1 200px; }
16
17 /* Arama cubugu: genis input + sabit buton */
18 .arama {
19   display: flex;
20   gap: 0.5rem;
21 }
22 .arama input { flex: 1; min-width: 0; }
23 .arama button { flex: none; }
24
25 /* Onay kutusu + etiket hizalama */
26 .onay {
27   display: flex;
28   align-items: center;
29   gap: 0.5rem;
30 }
```

✓ Neden Bu Kadar Kolay?

Form elemanlarının hizalanması, CSS'in tarihsel olarak **en zor** konularındandı. Flexbox ile:

- ▶ Etiket-input dikey hizalama: `column`
- ▶ Yan yana alanlar: `flex: 1 1 200px`
- ▶ Onay kutusu ortalama: `align-items: center`

⚠ Dikkat

```
.arama input { flex: 1;
min-width: 0; }      min-width: 0
```

şarttır — aksi halde `<input>`'in varsayılan genişliği (yaklaşık 20 karakter) onun küçülmesini engeller ve buton dışarı taşar.



Yapay Zeka ile Flexbox

BÖLÜM 4

Flexbox'ta Yapay Zeka Ne Kadar İyi?

👍 Genelde Doğru

- ▶ Standart kalıplar (ortalama, menü, kart ızgarası)
- ▶ `flex` kısa yazımını açıklama
- ▶ Eksen kavramını anlatma
- ▶ “Neden çalışmıyor?” teşhisi (kod verirsiniz)

👎 Sık Hataları

- ▶ **Flexbox yerine Grid gereken** yerde Flexbox önerme
- ▶ Gereksiz `width: 33.33%` hesapları (`flex: 1` varken)
- ▶ `gap` yerine eski `margin` hileleri
- ▶ `order` kullanıp erişilebilirliği bozma
- ▶ `flex-wrap: wrap` eklemeyi unutma

⚠ Dikkat

En kritik ayırt etme: “Bu iş Flexbox ile mi Grid ile mi yapılmalı?” sorusunda yapay zeka **sorulduğu şeyi** önerir. “Flexbox ile nasıl yaparım” diye sorarsanız, Grid daha uygun olsa bile size Flexbox çözümü üretir. **Açık uçlu sorun.**

Yapay Zeka İstemi (Prompt) — Flexbox mi Grid mi?

Bir kitap sitesi tasarliyorum. Asagidaki 8 duzen ihtiyacim var. Her biri icin FLEXBOX mi yoksa GRID mi kullanmam gerektigini soyle.

1) Ust barda solda logo, sagda menu 2) Menudeki 5 linkin yan yana dizilmesi 3) Ana sayfada 12 kitap kartinin duzenli izgara halinde 4) Sayfanin genel duzeni: header / sidebar / main / footer 5) Kart icinde: kapak, baslik, yazar, en altta buton 6) Bir butonun icinde ikon + metin hizalamasi 7) Form satiri: genis input + sabit genislikte buton 8) Fotograf galerisi: bazi gorseller 2 sutun genisliginde

Formatin: | # | Secim | Neden (tek cumle) | Anahtar CSS ozellikleri |

KURAL: "Ikisi de olur" deme. Her madde icin NET bir secim yap ve karari "tek boyut mu, iki boyut mu?" testine dayandir.

Cevapları **gelecek hafta Grid'i öğrendikten sonra** tekrar değerlendirin.

Yapay Zeka İstemi (Prompt) — “Öğelerim Neden Yan Yana Gelmiyor?”

Flexbox düzenim beklendiği gibi çalışmıyor. Aşağıda HTML ve CSS var.

GÖREV: 1) HANGİ element flex kapsayıcı? Doğru elemente mi uygulanmış? 2) Flex ÖGELERİ hangileri? (Sadece doğrudan çocuklar!) 3) Ana eksen ve çapraz eksen hangi yönde? 4) justify-content ve align-items doğru ekseninde mi kullanılmış? 5) flex-wrap gerekli mi? Eksikse ne olur?

Her tespit için "SORUN" veya "TAMAM" etiketi kullan. Düzeltilmiş tam kodu verme; sadece DEĞİSTİRİLECEK satırları "eski -> yeni" formatında göster.

HTML: [KODUNUZ]

CSS: [KODUNUZ]

2. madde kritik: Flexbox'ta en sık hata, **torun** elemanların flex ögesi sanılmasıdır.

Yapay Zeka İstemi (Prompt) — Flexbox Kalıp Kartı

Bana bir "Flexbox Kalip Kutuphanesi" hazirla. Asagidaki 10 yaygin duzen ihtiyaci icin en KISA ve MODERN CSS cozumunu ver.

1) Dikey + yatay tam ortalama 2) Ust bar: sol logo, sag menu 3) Duyarli kart izgarasi (media query KULLANMADAN) 4) Yapiskan footer 5) Kart icinde butonu en alta itme 6) Ikon + metin satir ici hizalama 7) Esit genislikte 3 sutun 8) Sabit sidebar + esneyen ana icerik 9) Son ogeyi saga itme 10) Ogeler arasi ayirici cizgi (border ile, ilk/son haric)

HER cozum icin: - Sadece gerekli CSS satirlari (en fazla 6 satir) - "gap" kullan, margin hilelerinden KACIN - Tek cumlelik aciklama

Sonunda: bu 10 cozumden hangilerinin GRID ile daha iyi yapilabilecegini isaretle.

Özet ve Değerlendirme

BÖLÜM 5

Eksenler

`justify-content` → **ana eksen**
`align-items` → **çapraz eksen**
`flex-direction` eksenleri **değiştirir**.

Esneklik

`flex: grow shrink basis`.
`flex: 1` = eşit sütun.
`flex: 1 1 260px` = duyarlı kart.

Boşluk

`gap` **kullanın**, `margin` hilelerine gerek yok. Kart ızgarasında `flex-wrap: wrap` **unutmayın**.

Yapay Zeka

Standart kalıplarda güvenilir. Ama **“Flexbox ile nasıl?”** diye sorarsanız, Grid daha uygun olsa bile Flexbox önerir — **açık uçlu sorun**.

Altın Kural: **Tek boyut** (bir satır veya bir sütun) → Flexbox. **İki boyut** → Grid (gelecek hafta).

Terim	Anlam	Terim	Anlam
Flex container	<code>display: flex</code> olan ana element	<code>flex-grow</code>	Boşluk payı oranı
Flex item	Kapsayıcının doğrudan çocuğu	<code>flex-shrink</code>	Küçülme oranı
Ana eksen	<code>flex-direction</code> yönü	<code>flex-basis</code>	Başlangıç boyutu
Çapraz eksen	Ana eksene dik yön	<code>align-self</code>	Tek öğe hizalaması
<code>justify-content</code>	Ana eksen hizalama	<code>order</code>	Görsel sıra (DOM'u değiştirmez)
<code>align-items</code>	Çapraz eksen hizalama	<code>gap</code>	Öğeler arası boşluk
<code>flex-wrap</code>	Satıra sığmayanların davranışı	Sticky footer	Footer'ı en altta tutma

1 `flex-direction: column` olan bir kapsayıcıda `justify-content: center` ne yapar? `align-items: center` ne yapar?

2 Aşağıdaki kodda öğeler alt satıra geçmiyor, eziliyor. Neden?

```
.kartlar { display: flex; gap: 1rem; }  
.kart    { flex: 1 1 300px; }
```

3 `flex: 1`, `flex: auto` ve `flex: none` kısa yazımlarının açık karşılıklarını yazınız.

4 Bir kartın içindeki butonu **her zaman kartın en altında** tutmak için ne yaparsınız?

5 `order: -1` kullanmanın erişilebilirlik riski nedir?

6 Bir `<div>` içinde `<span>` var, o `<span>` içinde de `<a>` var. `<div>`'e `display: flex` verdiniz. `<a>` bir flex ögesi olur mu? Neden?

Laboratuvar Föyü 8

- ▶ BookVerse yatay menüsü
- ▶ `justify-content` değer tablosu (görsel)
- ▶ Duyarlı kart ızgarası (media query'siz)
- ▶ Kart içi dikey flex + alta itilmiş buton
- ▶ Yapışkan footer
- ▶ DevTools Flexbox denetçisi kullanımı
- ▶ flexboxfroggy.com 24 seviye
- ▶ Yapay zeka araç seçimi denetimi

Hazırlık

Okuma: Kitap Ünite 2.2 — “Modern CSS Düzenleri (Flexbox)” **Kaynaklar:**

- ▶ css-tricks.com/snippets/css/a-guide-to-flexbox/
- ▶ flexboxfroggy.com
- ▶ developer.mozilla.org/docs/Web/CSS/CSS_Flexbox

Gelecek hafta: **Grid** ile iki boyutlu düzen ve duyarlı tasarım.

Haftaya Görüşmek Üzere!

Gelecek Hafta:

CSS Grid ve Duyarlı Tasarım

- birimi ▪
- Media query ▪ Mobile-first yaklaşım