

# Bab 8

## CSS BOX MODEL, FLEXBOX, GRID

---

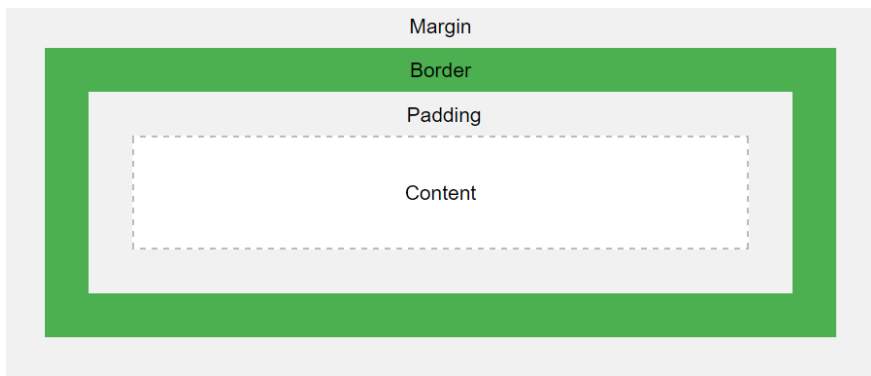
### Bab ini membahas:

- CSS Box Model
  - CSS Flexbox
  - CSS Grid
- 

### CSS Box Model

Semua elemen HTML dapat dianggap sebagai kotak. Dalam CSS, istilah “Box Model” digunakan ketika berbicara tentang desain dan tata letak [2].

Model kotak CSS pada dasarnya adalah kotak yang membungkus setiap elemen HTML. Ini terdiri dari: margin, border, padding, dan konten yang sebenarnya.



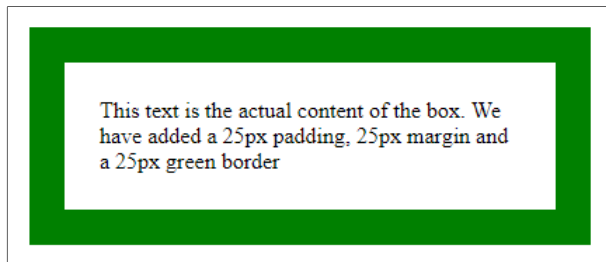
**Gambar 10** CSS Box Model

Penggunaan Div digunakan untuk membuat Box Model. Sebagai contoh tuliskan deklarasi *style* berikut pada file HTML dengan nama **box\_model.html**.

```
<html>
<head>
<title>CSS Box Model</title>
<style type="text/css">
div {
    width: 300px;
    border: 25px solid green;
    padding: 25px;
    margin: 25px;
}
</style>
</head>

<body>
<div>This text is the actual content of the box. We
have added a 25px padding, 25px margin and a 25px
green border </div>
</bod>
</html>
```

Jalankan file **box\_model.html**, hasil tampilan dapat dilihat pada gambar di bawah ini.



**Gambar 11 CSS Box Model**

## CSS Flexbox

*The Flexible Box (Flexbox) Layout Module*, digunakan untuk memudahkan merancang struktur tata letak responsif yang fleksibel tanpa menggunakan *float* atau *positioning* [2].

Flexbox Model dimulai dengan membuat *class* `flex-container` yang berkedudukan sebagai *parent* dengan menggunakan *property* `display` bernilai `flex`. Selanjutnya elemen `<div>` sebagai area dari konten dideklarasikan sebagai *child*, dengan *property* yang unik.

Sebagai contoh pada dokumen HTML penerapan Flexbox dilakukan pada elemen `<div>` dengan menggunakan *selector* `class flex-container`.

```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>
```

Selanjutnya `flex-container` menjadi fleksibel dengan mengatur *property* `display` dengan nilai `flex`.

```
.flex-container {
  display: flex;
}
```

Selain *property* `display`, *property* lainnya yang dapat digunakan antara lain `flex-direction`, `flex-wrap`, `flex-flow`, `justify-content` (penjelasan properti-properti tersebut setelah contoh lengkap di bawah ini).

Buat file dengan nama **flexbox.html** dengan menuliskan *source code* di bawah ini.

```
<html>
<head>
<style>
  .flex-container {
    display: flex;
    background-color: DodgerBlue;
  }
```

```
.flex-container > div {  
  background-color: #f1f1f1;  
  margin: 10px;  
  padding: 20px;  
  font-size: 30px;  
}  
</style>  
</head>  
<body>  
<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>  
</div>  
</body>  
</html>
```



**Gambar 12** Hasil Penerapan Flexbox Model

Properti-properti lain pada Flexbox yang dapat digunakan antara lain:

- **flex-direction**

Digunakan untuk menentukan ke arah mana kontainer akan menumpukkan item-item fleksibelnya. Nilai dari *flex-direction* dapat berisi *column* (menampilkan item-item fleksibel secara *vertical* dari *top* ke *bottom*), *column-reverse* (menampilkan item-item fleksibel secara *vertical* terbalik dari *bottom* ke *top*), *row* (menampilkan item-item fleksibel secara *horizontal* dari *left* ke *right*), *row-reverse* (menampilkan item-item fleksibel secara *horizontal* terbalik dari *right* ke *left*). Contoh penerapan *flex-direction*.

```
.flex-container {  
  display: flex;
```

```
flex-direction: row;  
}
```

## The flex-direction Property

The "flex-direction: row;" stacks the flex items horizontally (from left to right):



**Gambar 13** Hasil Penggunaan flex-direction Dengan Nilai *row*

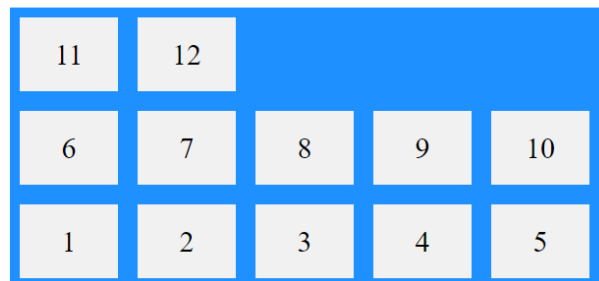
- **flex-wrap**

Properti flex-wrap digunakan untuk menentukan apakah item-item flex perlu dibungkus atau tidak. Nilai yang dapat digunakan pada properti ini meliputi wrap (membungkus item-item fleksibel), nowrap (tanpa membungkus item-item fleksibel), wrap-reverse (membungkus item-item fleksibel secara terbalik). Contoh penggunaan flex-wrap:

```
.flex-container {  
  display: flex;  
  flex-wrap: wrap-reverse;  
}
```

## The flex-wrap Property

The "flex-wrap: wrap;" specifies that the flex items will wrap if necessary:



**Gambar 14** Hasil Penggunaan flex-wrap Dengan Nilai *wrap-reverse*

- **flex-flow**

Properti flex-flow digunakan untuk meringkas / menggabungkan nilai dari properti flex-direction dan flex-wrap. Sebagai contoh:

```
.flex-container {
  display: flex;
  flex-flow: row wrap;
}
```

- **justify-content**

Properti justify-content digunakan untuk meratakan item-item fleksibel. Nilai dari properti ini dapat bernilai center (meratakan item fleksibel ke tengah), flex-start (meratakan item-item fleksibel di awal wadah), flex-end (meratakan item-item fleksibel di akhir wadah), space-around (membuat item-item fleksibel dengan spasi sebelum, di antara, dan setelah baris sama rata), space-between (menampilkan item-item fleksibel dengan ruang / *space* di antara garis). Nilai flex-start merupakan nilai default / bawaan.



**Gambar 15** Hasil Penggunaan justify-content Dengan Nilai space-around



**Gambar 16** Hasil Penggunaan justify-content Dengan Nilai space-between

### Latihan 3 – Membuat Gallery Foto

Buatlah *gallery* foto menggunakan CSS Flexbox dengan 9 (sembilan) item foto dengan dimensi 3 x 3 (tiga kolom dan tiga baris)!

## CSS Grid

*The CSS Grid Layout Module* digunakan untuk memudahkan merancang struktur tata letak berbasis grid, dengan baris dan kolom, membuatnya lebih mudah untuk mendesain halaman web tanpa menggunakan *float* dan *positioning* [2]. Layout grid terdiri dari elemen *parent* dengan satu atau lebih elemen anak (*child*).

Elemen HTML menjadi wadah (*container*) dengan cara property `display` diisi dengan nilai `grid` atau `inline-grid`. Dengan menerapkan nilai `grid` atau `inline-grid`, maka semua `grid container` akan menjadi `grid item`.

Penggunaan nilai <code>grid</code>	Penggunaan nilai <code>inline-grid</code>
<pre>.grid-container {   display: grid; }</pre>	<pre>.grid-container {   display: inline-grid; }</pre>

Secara lengkap penggunaan layout dengan grid pada dokumen HTML sebagai berikut. Buatlah dengan nama file **gridlayout.html**.

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
  .grid-container {  
    display: grid;  
    grid-template-columns: auto auto auto;  
    background-color: #2196F3;  
    padding: 10px;  
  }  
  .grid-item {  
    background-color: rgba(255, 255, 255, 0.8);  
    border: 1px solid rgba(0, 0, 0, 0.8);  
    padding: 20px;  
    font-size: 30px;  
    text-align: center;  
  }  
</style>
```

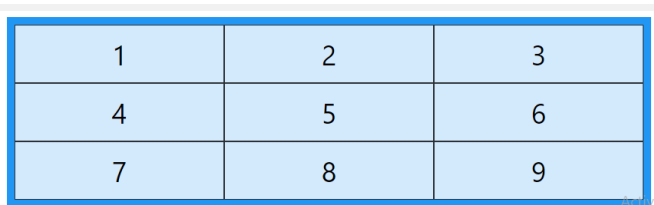
```

</head>
<body>

<h1>The display Property:</h1>

<div class="grid-container">
  <div class="grid-item">1</div>
  <div class="grid-item">2</div>
  <div class="grid-item">3</div>
  <div class="grid-item">4</div>
  <div class="grid-item">5</div>
  <div class="grid-item">6</div>
  <div class="grid-item">7</div>
  <div class="grid-item">8</div>
  <div class="grid-item">9</div>
</div>
</body>
</html>

```



1	2	3
4	5	6
7	8	9

**Gambar 17** Hasil Penerapan Grid

Properti-properti yang dapat digunakan pada CSS Grid untuk mengatur kolom dan baris meliputi `grid-column-gap` (mengatur jarak antar kolom), `grid-row-gap` (mengatur jarak antar baris), `grid-gap` (digunakan untuk memperingkas penulisan pengaturan kolom dan baris yaitu berisi nilai kolom dilanjutkan baris). Berikut contoh penerapan `grid-column-gap`:

```

.grid-container {
  display: grid;
  grid-column-gap: 50px;
}

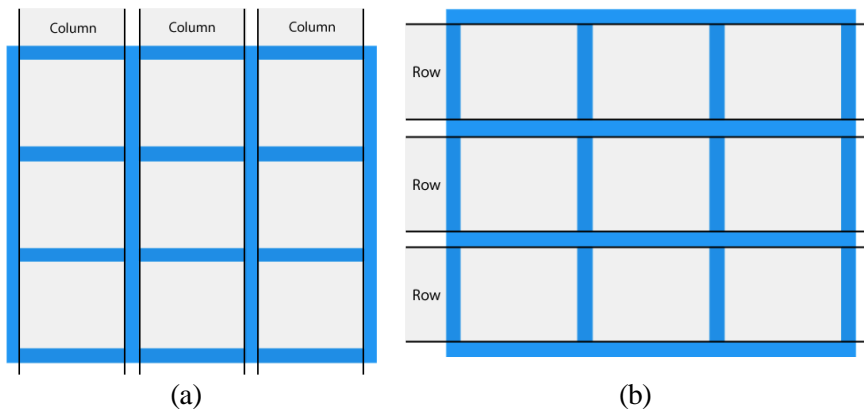
```



1	2	3
4	5	6
7	8	9

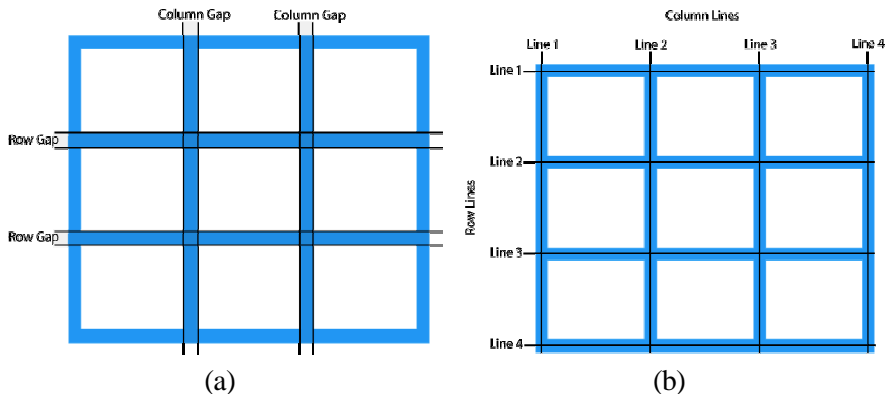
**Gambar 18** Hasil Penerapan `grid-column-gap` Dengan Nilai 50px

Gambar di atas menjelaskan penerapan property `grid-column-gap` sebesar 50px, maka kolom yang telah tersusun akan memiliki jarak (*gap*) sebesar 50px. Untuk lebih jelasnya, pada gambar di bawah ini telah disajikan konsep kolom dan baris.



**Gambar 19** Konsep Kolom / *Column* (a) dan Baris / *Row* (b)

Selain konsep kolom dan baris, konsep *gap* dan *line* juga perlu dipahami. Pada gambar di bawah ini disajikan perbedaan antara *gap* dengan *line*.



**Gambar 20** Perbedaan Konsep *Gap* (a) Dengan *Line* (b)

### Grid Gap

Properti `grid-gap` dapat menggunakan satu nilai saja yang akan mewakili isi dari gap kolom dan baris. Sebagai contoh:

```
.grid-container {
  display: grid;
  grid-gap: 50px;
}
```

Pada contoh di atas nilai yang diisi pada properti `grid-gap` adalah 50px yang merupakan ringkasan dari deklarasi `grid-column-gap: 50px` dan `grid-row-gap: 50px`.

### Grid Line

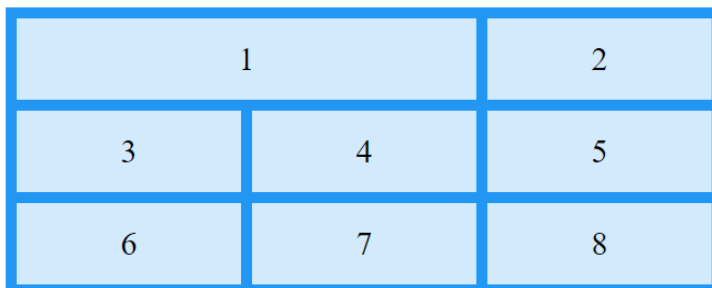
*Line* yang berada di antara kolom disebut dengan *column lines*. Sedangkan *line* yang berada di antara baris disebut dengan *row lines*. Sebagai contoh buatlah CSS dengan *class* `item1` seperti berikut:

```
.item1 {
  grid-column-start: 1;
  grid-column-end: 3;
}
```

Jika *class* *item1* yang telah dideklarasikan sebelumnya, diterapkan pada dokumen HTML dengan item-item yang ada sebanyak 8 (delapan) item seperti di bawah ini.

```
<div class="grid-container">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
  <div class="item5">5</div>
  <div class="item6">6</div>
  <div class="item7">7</div>
  <div class="item8">8</div>
</div>
```

Maka item angka 1 akan memiliki proporsi yang lebih lebar disebabkan *grid-column-end* berakhir pada line ke-3 atau bernilai 3 (tiga).



1		2
3	4	5
6	7	8

**Gambar 21** Hasil Item 1 Dengan *grid-column-end* Bernilai 3 (Tiga)

### Grid Container

Grid Container berfungsi sebagai wadah bagi *grid items* yang ditempatkan di dalam kolom dan baris. *Grid-template-columns* merupakan properti yang digunakan untuk mengatur jumlah kolom pada *grid layout* dan lebar setiap kolom. Sebagai contoh jika kita menginginkan *grid layout* sejumlah 4 (empat) kolom dan kita mengatur lebar (*width*) untuk setiap kolom adalah sama. Maka kita cukup memberikan nilai *auto*.

```
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto auto;
}
```

1	2	3	4
5	6	7	8

**Gambar 22** Hasil Penggunaan `grid-template-column` 4 Kolom Dengan Nilai *Auto*

Jika ingin mengatur lebar yang berbeda pada kolom-kolom *grid layout*, dapat mengubah nilai *auto* dengan nilai angka dalam satuan pixel (px). Sebagai contoh.

```
.grid-container {
  display: grid;
  grid-template-columns: 80px 200px auto 30px;
}
```

Pada *script* di atas kolom pertama memiliki lebar 80px, kolom kedua 200px, kolom ketiga menyesuaikan, dan kolom keempat 30px.

1	2	3	4
5	6	7	8

**Gambar 23** Hasil Penggunaan `grid-template-column` 4 Kolom Berbeda Lebar

Selain pengaturan pada kolom, grid layout dapat juga diatur berdasarkan baris (*row*) dengan menggunakan properti `grid-template-rows` dengan pemberian nilai yang sama seperti kolom.

### Grid Item

Secara default setiap grid container memiliki satu grid item untuk setiap kolom pada setiap baris. Namun kita dapat mengatur sendiri grid item, misalkan dengan menggabungkan beberapa kolom menjadi satu. Sebagai contoh jika `item1` merupakan penggabungan beberapa kolom dimulai dari line 1 (satu) dan berakhir pada line 5 (lima) dapat kita tulis.

```
.item1 {  
  grid-column: 1 / 5;  
}
```

Penggunaan properti `grid-column` merupakan cara ringkas dari penggunaan properti `grid-row-start` dan `grid-row-end`.

1				2	3
4	5	6	7	8	9
10	11	12	13	14	15

**Gambar 24** Hasil Penggunaan `grid-column` Dengan Nilai `1/5`

*Span* dapat kita terapkan dalam pengisian nilai pada properti `grid-column`. Sehingga `grid-column: 1 / 5` sama dengan `grid-column: 1 / span 4`.

Selain `grid-column`, terdapat properti `grid-row` untuk mengatur baris dimana item akan dimulai (*start*) dan dimana item akan berakhir (*end*) pada *grid layout*. Sebagai contoh:

```
.item1 {  
  grid-row: 1 / 4;  
}
```

1	2	3	4	5	6
	7	8	9	10	11
	12	13	14	15	16

**Gambar 25** Hasil Penggunaan `grid-row` Bernilai 1 / 4

`grid-row: 1 / 4` memiliki kesamaan jika kita menggunakan *span* dengan menulis `grid-row: 1 / span 3`.

#### Latihan 4 – Membuat Layout Menggunakan CSS Grid

Buatlah layout seperti pada gambar di bawah ini dengan menggunakan CSS Grid!

Header		
Menu	Main	Right
	Footer	