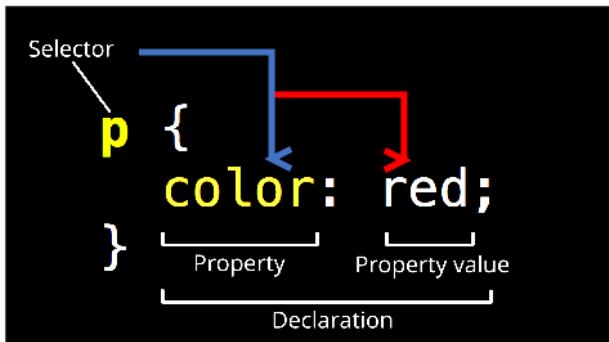


# Anatomi CSS

## Anatomi CSS



### • Selector

Apa yang ingin dimodifikasi

### • Property

Bagian apa yang ingin dimodifikasi

### • Property Value

Bentuk modifikasinya seperti apa

Contoh di atas, yang ingin dimodifikasi adalah seluruh tag `<p>` pada komponen **warna teksnya** menjadi **warna merah**

## Percobaan pertama

### Penjelasan

`<title>` : Menetapkan judul dokumen yang akan ditampilkan di tab browser.

`<style>` : Memulai blok kode CSS untuk mengatur tampilan elemen HTML.

`p { color: red; }` : Mengatur teks pada elemen paragraf `<p>` menjadi berwarna merah

`<p>Walcome CSS!</p>` : Dua paragraf dengan teks "Walcome CSS!" yang akan mewarisi gaya teks yang telah diatur dalam blok CSS sebelumnya.

## Kode program

```
<!DOCTYPE html>
<html>
  <head>
    <title>Percobaan Pertama CSS</title>
    <style>
      p {
        color: red;
      }
    </style>
  </head>
```

```
<body>
  <p>Walcome CSS!</p>
  <p>Walcome CSS!</p>
</body>
</html>
```

## Hasil



Walcome CSS!

Walcome CSS!

## Percobaan kedua

### kode CSS

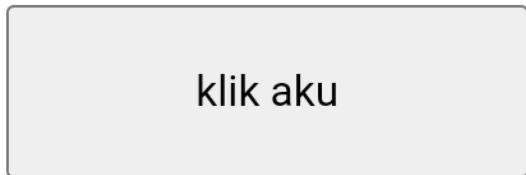
```
button {
  width: 150px;
  height: 50px;
  background-color: aqua;
  font-weight: bold;
  font-size: 30px;
}
```

### background-color

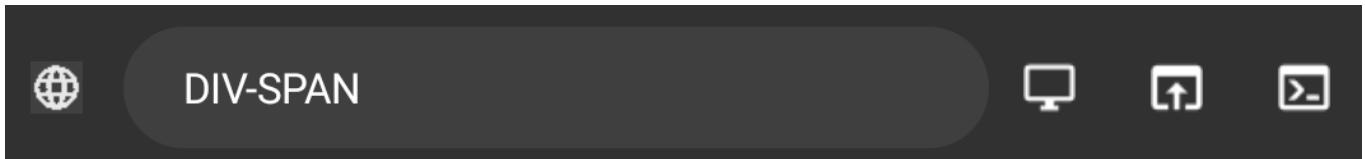
**Before:**



WELCOME CSS!



**After:**



WELCOME CSS!



**font-weight**

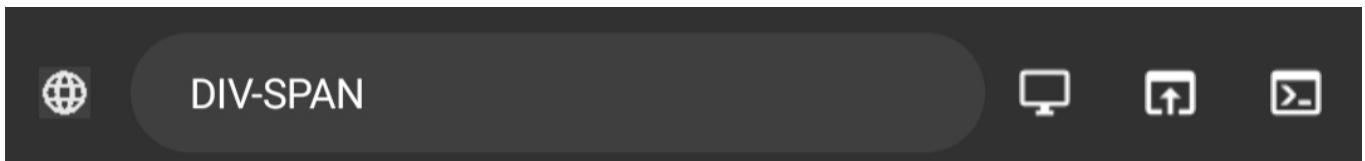
**Before:**



WELCOME CSS!

klik aku

**After:**



WELCOME CSS!

klik aku

**font-size**

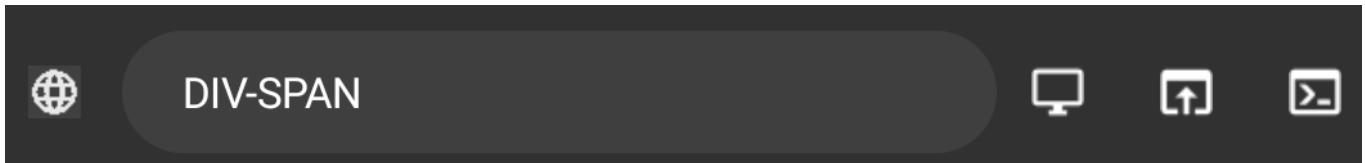
**Before:**



WELCOME CSS!

klik aku

**After:**



WELCOME CSS!

klik aku

## Cara pemanggilan CSS

### Inline

Pemanggilan inline CSS adalah cara untuk menambahkan gaya langsung ke elemen HTML menggunakan atribut `style`. Ini memungkinkan Anda menentukan gaya khusus untuk elemen tertentu tanpa perlu membuat file CSS terpisah.

**contoh kode program:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>belajar panggilan css</title>
  </head>
  <body>
    <p style="color:red;">inline</p>
  </body>
</html>
```

## Internal

Pemanggilan internal CSS adalah ketika Anda menulis gaya CSS di dalam tag `style` di bagian `head` dari dokumen HTML yang sama

### contoh kode program:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Percobaan Pertama CSS</title>
    <style>
      p {
        color: red;
      }
    </style>
  </head>
  <body>
    <p>Walcome CSS!</p>
    <p>Walcome CSS!</p>
  </body>
</html>
```

## External

Pemanggilan eksternal CSS adalah ketika Anda menyimpan gaya CSS dalam file terpisah dengan ekstensi `css` dan memanggilnya dalam dokumen HTML menggunakan tag `link`. Ini memungkinkan Anda untuk memisahkan struktur HTML dari gaya CSS, membuat kode lebih terorganisir dan mudah dikelola

### contoh kode program:

```
<!DOCTYPE html>
<html>
<head>

    <title>CSS</title>
    <link rel="stylesheet" href="Contoh.css">
</head>
<body>

<p>menggunakan pemanggilan external</p>

</body>
</html>
```

# Selector

## Elemen selector

Selector ini memilih semua elemen HTML dengan nama elemennya. Misalnya, jika Anda menggunakan `P` sebagai selector, maka semua elemen paragraf dalam dokumen HTML akan dipilih

## Class selector

digunakan untuk menerapkan gaya pada elemen HTML yang memiliki kelas tertentu. Ini memungkinkan Anda untuk mengatur gaya tertentu untuk kelompok elemen yang memiliki kelas yang sama, tanpa harus merubah setiap elemen secara individual. Untuk menggunakan pemilih kelas, Anda menambahkan titik (.) diikuti oleh nama kelas yang ingin Anda targetkan dalam aturan gaya CSS Anda

## ID selector

Memilih elemen berdasarkan ID uniknya. Untuk menggunakan selector ID, Anda harus menambahkan tanda pagar (#) di depan nama ID. Contoh: `#id` akan memilih elemen dengan ID "id"

# Materi Text

## Text-align

### Penjelasan:

Properti text-align menentukan perataan horizontal teks dalam sebuah elemen.

`left` : Alirkan teks ke kiri.

`right` : Alirkan teks ke kanan.

`center` : Alirkan teks ke tengah.

`justify` : Meratakan teks ke kiri dan kanan, dengan menyesuaikan spasi antarkata untuk mengisi lebar elemen.

`justify-all` : Sama seperti justify, namun juga meratakan spasi antarkata pada baris terakhir.

## **kode program:**

```
p{text-align: center;}
```

## **Hasil:**



## **kesimpulan:**

Dengan lima nilai yang umum digunakan, yaitu left, right, center, justify, dan justify-all, properti ini memberikan fleksibilitas dalam menyesuaikan posisi teks sesuai dengan kebutuhan desain halaman web.

## **Text-decoration**

### **Penjelasan**

Kode CSS `p {text-decoration: underline;}` digunakan untuk memberikan dekorasi garis bawah pada teks dalam elemen paragraf (p) di halaman web. Dengan demikian, semua teks di dalam elemen paragraf akan memiliki garis bawah.

## **Kode program**

```
p{Text-decoration:underline;}
```

## **Hasil**

←

→

⟳

🔍

ⓘ

File

C:/pemro

## contoh CSS

## Kesimpulan

Kesimpulannya, kode CSS `p {text-decoration: underline;}` digunakan untuk memberikan dekorasi garis bawah pada teks dalam elemen paragraf (`<p>`) di halaman web.

## Text-Transform

### penjelasan

Kode CSS `p {text-transform: lowercase;}` digunakan untuk mengubah semua teks dalam elemen paragraf (p) menjadi huruf kecil (lowercase). Ini berarti semua huruf dalam teks akan ditampilkan dalam bentuk huruf kecil.

### program

```
p{text-transform:lowercase;}
```

### hasil

←

→

⟳

🔍

ⓘ

contoh css

## kesimpulan

Kesimpulannya, kode `{text-transform:lowercase;}` ini akan membuat semua teks di dalam elemen paragraf menjadi huruf kecil.

## Text-indent

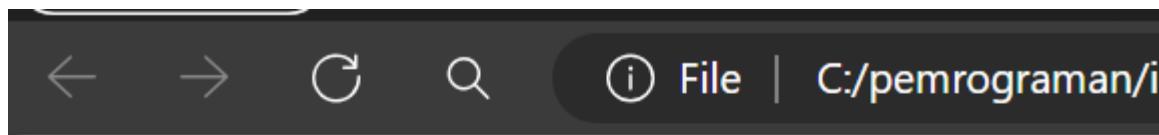
## penjelasan

text-indent: Ini adalah properti CSS yang mengatur jarak indentasi awal dari teks dalam sebuah elemen. 50px: Ini adalah nilai indentasi yang diberikan dalam piksel. Dalam kasus ini, teks dalam semua elemen paragraf akan di-indentasi sejauh 50 piksel dari sisi kiri.

## program

```
p{Text-indent:50px;}
```

## hasil



contoh CSS

## kesimpulan

Kesimpulannya, kode tersebut mengatur indentasi awal teks dalam semua elemen paragraf sejauh 50 piksel dari sisi kiri.

## Letter-spacing

### penjelasan

{Letter-spacing: 20px;} mengatur jarak antara baris dalam elemen paragraf (p) menjadi 20 piksel. Ini berarti setiap baris teks dalam elemen paragraf akan memiliki jarak horizontal sebesar 20 piksel.

## program

```
p{letter-spacing:20px;}
```

## hasil

c o n t o h C S S

## kesimpulan

Kesimpulannya, kode CSS ini akan membuat jarak antara baris dalam elemen paragraf menjadi 20 piksel, menciptakan ruang horizontal yang lebih besar di antara setiap baris teks.

## Line-height

### penjelasan

line-height: Ini adalah properti CSS yang mengatur tinggi baris di dalam elemen. Di sini, nilainya adalah 50px, yang berarti jarak antara baris dalam paragraf akan setara dengan 50 piksel.

## program

```
P{Line-height:50px;}
```

## hasil

contoh CSS

## **kesimpulan**

kode Line-height:170px; mengatur tinggi baris untuk semua elemen paragraf menjadi 170 piksel.

## **Word-Spacing**

### **penjelasan**

word-spacing: Ini adalah properti CSS yang mengatur jarak antara kata-kata di dalam elemen. Di sini, nilainya adalah 50px, yang berarti jarak antara kata-kata dalam paragraf akan setara dengan 50 piksel.

### **program**

```
P{Word-Spacing: 50px ;}
```

### **hasil**



contoh CSS

## kesimpulan

kode Word-spacing:50px; mengatur jarak antara kata-kata di dalam semua elemen paragraf menjadi 50 piksel.

---

## Materi Font

### Font-Weight

#### penjelasan

Font-weight: bold; adalah properti CSS yang digunakan untuk mengatur ketebalan teks. Dalam hal ini, nilai yang diberikan adalah "bold", yang membuat teks yang menggunakan properti ini untuk diatur dengan ketebalan yang lebih besar dari teks biasa.

#### program

```
p{Font-Weight:bold;}
```

#### hasil



contoh CSS

## kesimpulan

Properti CSS ini digunakan untuk mengatur ketebalan teks. Dalam hal ini, nilai yang diberikan adalah "bold".

## Font-Size

### penjelasan

Font-size: 50px; adalah properti CSS yang digunakan untuk mengatur ukuran font menjadi 50 piksel. Ini berarti teks yang diberikan akan ditampilkan dengan ukuran 50 piksel, membuatnya lebih besar dari ukuran font standar yang biasa digunakan dalam tata letak halaman web

### program

```
P{Font-Size:50px;}
```

### hasil



## contoh CSS

### kesimpulan

Kode property bertujuan agar teks membuatnya lebih besar dari ukuran font standar yang biasa digunakan dalam tata letak halaman web.

## Font-Style

### penjelasan

Font-style: italic; adalah properti CSS yang digunakan untuk mengatur gaya teks menjadi miring (italic). Ini berarti teks yang diberikan akan miring, seperti yang sering kita lihat dalam

gaya teks untuk menekankan kata-kata atau membuat teks menonjol.

## program

```
p{Font-Style:italic;}
```

## hasil



*contoh CSS*

## kesimpulan

Kode properti yang bertujuan untuk mengatur gaya teks yang dalam kasus ini menjadi miring (italic).

## Font-family

### penjelasan

Font-family: Times New Roman; adalah properti CSS yang digunakan untuk menentukan jenis font yang akan digunakan untuk menampilkan teks. Dalam hal ini, jenis font yang dipilih adalah "Times New Roman." Ini akan mengubah teks yang menggunakan properti ini untuk diatur dengan gaya huruf yang khas dari jenis font Times New Roman, yang sering kali terlihat formal dan terstruktur.

## program

```
P{Font-family:Times-New-Roman;}
```

## hasil



## contoh CSS

### kesimpulan

Kode properti ini digunakan untuk menetapkan jenis font yang akan digunakan untuk menampilkan teks, di mana dalam kasus ini, jenis font yang dipilih adalah Times New Roman.

---

## Materi Back Ground

### Background-Size

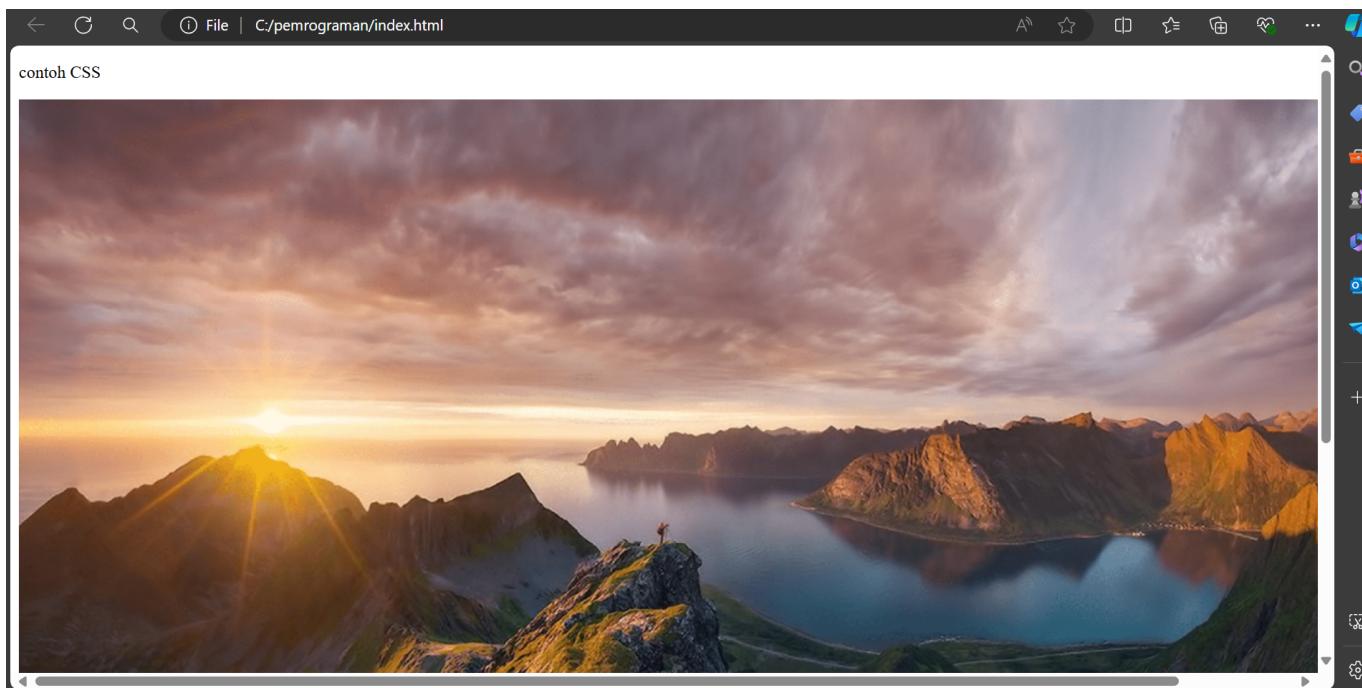
#### penjelasan

background-size: 200px; Properti ini mengatur ukuran latar belakang. Di sini, latar belakang akan memiliki lebar 200 piksel, dengan tinggi yang disesuaikan agar proporsi aslinya tetap terjaga.

#### program

```
P{background-size:200px;}
```

#### hasil



## kesimpulan

Kode property tersebut bertujuan mengatur ukuran gambar latar belakang dengan size pixel.

## Background-Repeat

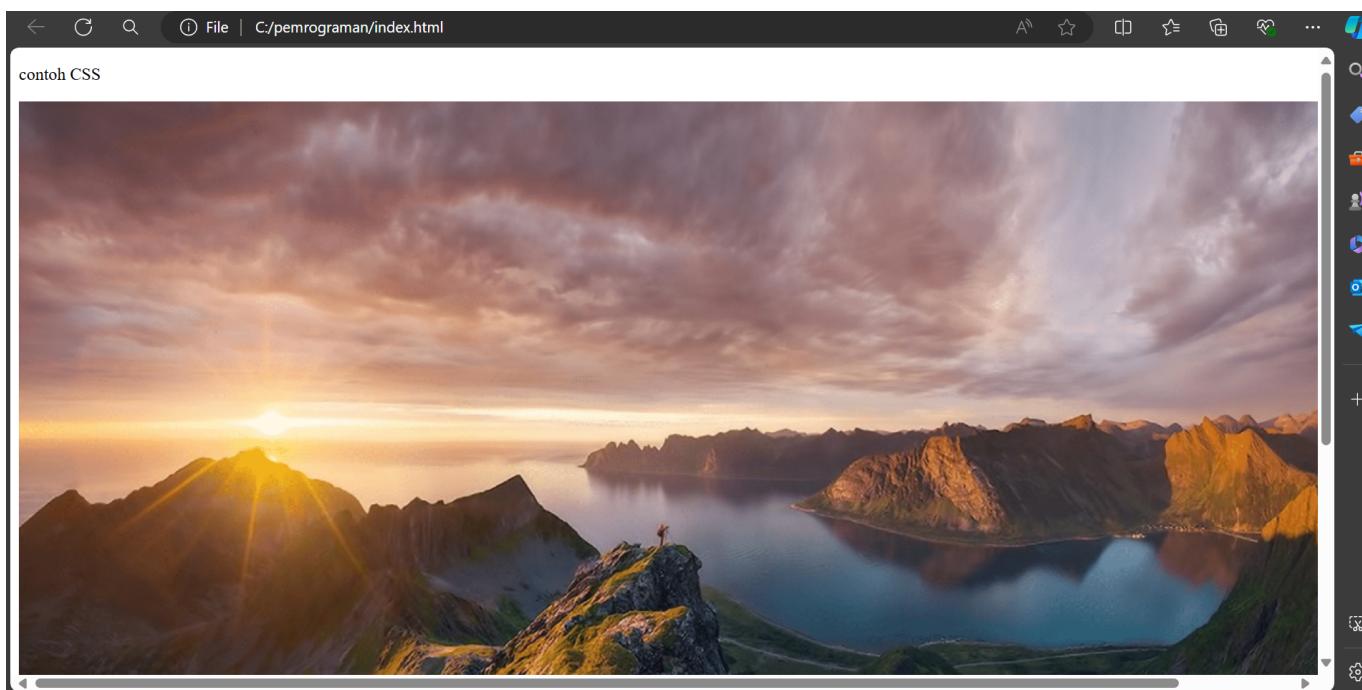
### penjelasan

background-repeat: no-repeat; Properti ini menentukan apakah gambar latar belakang akan diulang atau tidak. Dalam kasus ini, gambar latar belakang tidak akan diulang.

### program

```
P{background-repeat: no-repeat;}
```

### hasil



## kesimpulan

Kode property tersebut berguna agar gambar tidak akan ber ulang Dengan menggunakan value (no-repeat).

## Background-attachment

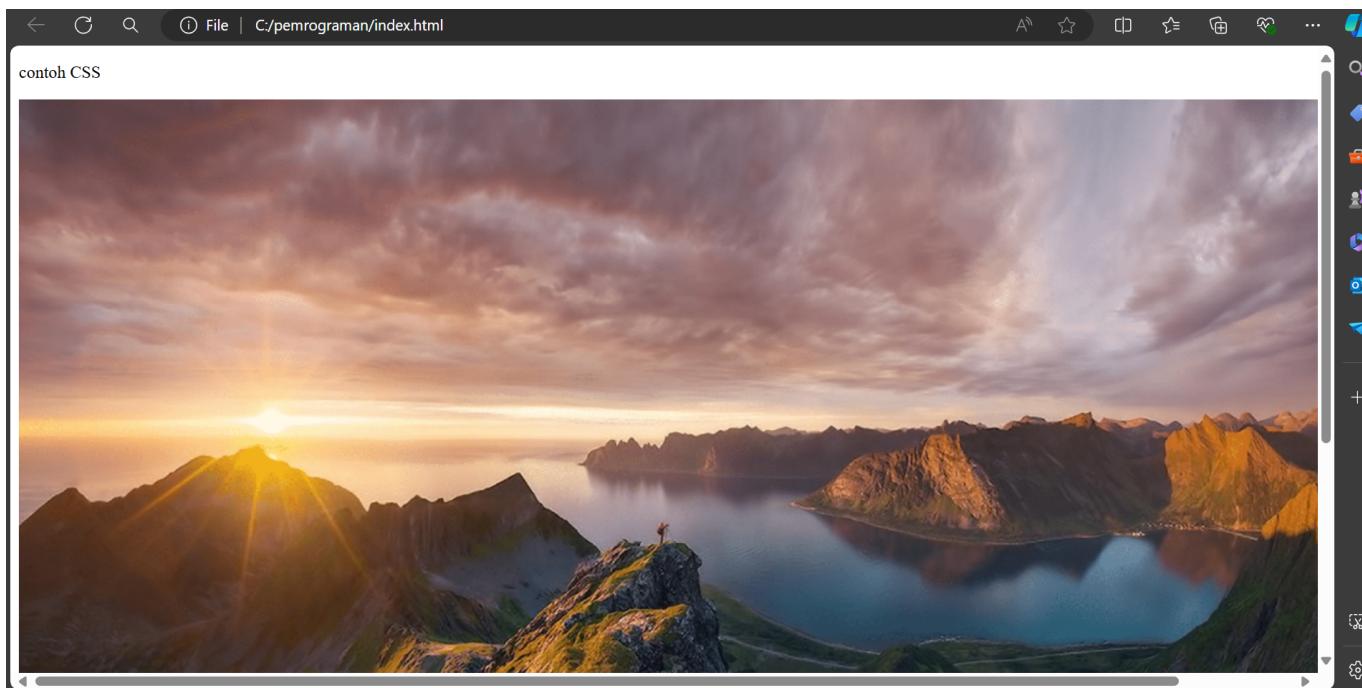
### penjelasan

background-attachment: fixed; Properti ini menentukan apakah latar belakang akan tetap diam atau akan bergulir bersama dengan isi elemen saat pengguna menggulir halaman. Dalam kasus ini, latar belakang akan tetap diam, artinya posisinya akan tetap konstan saat halaman digulir.

### program

```
p{background-attachment:fixed;}
```

### hasil



## kesimpulan

Kode property tersebut akan mengatasi gambar yang bergulir akan diam dengan menggunakan value (fixed)

## Background-position

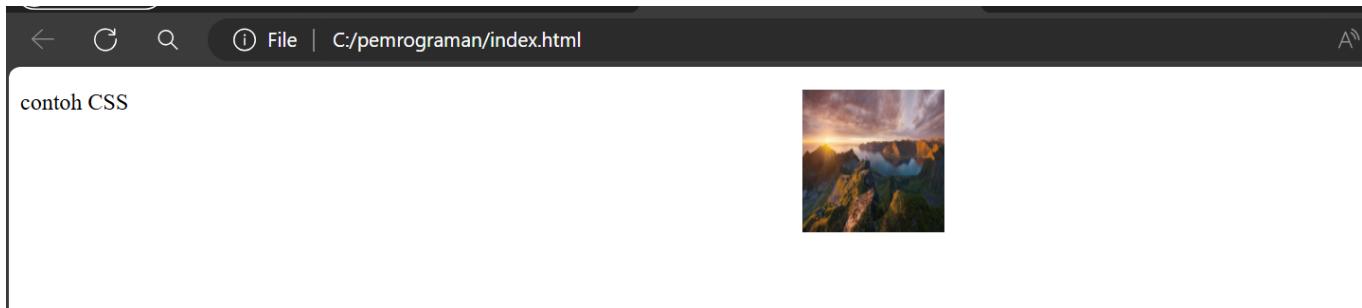
### penjelasan

background-position: top; Properti ini mengatur posisi latar belakang di dalam elemen. Di sini, latar belakang akan diposisikan di bagian atas elemen.

### program

```
p  
{Background-position:top;}
```

## hasil



## **kesimpulan**

Kode property tersebut akan menentukan posisi gambar latar belakang ke atas (top).

---

## **Materi Box Model**

### **width-height**

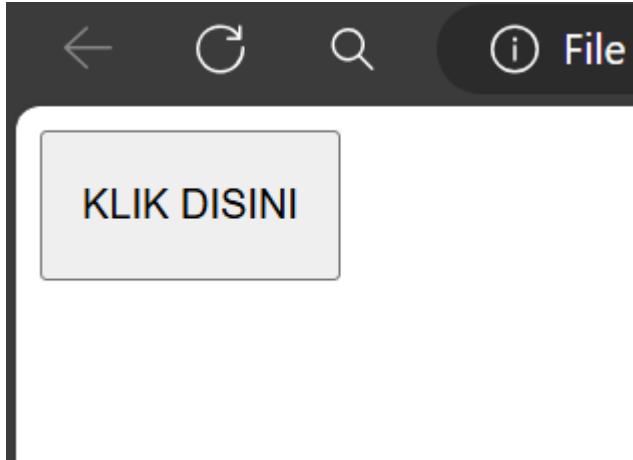
#### **penjelasan**

Jadi width dan height ini akan mengatur tinggi dan lebarnya dari Bordernya.

#### **program**

```
button {  
    width: 100px;  
    height: 50px;  
}
```

#### **hasil**



## **kesimpulan**

Kesimpulannya untuk tingginya ( height ) adalah 50px,dan lebarnya ( width ) adalah 100px

### **Border-width**

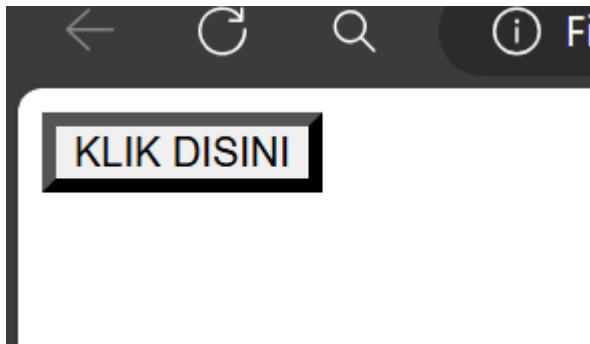
#### **penjelasan**

Untuk mengatur lebar border pada button.

## program

```
button {  
    border-width:5px;  
}
```

## hasil



## kesimpulan

mengatur lebar border

## Border-radius

### penjelasan

Border radius adalah properti CSS yang digunakan untuk mengatur sudut lengkung pada sudut-sudut elemen yang memiliki batas (border).

## program

```
button {  
    border-radius:10px 10px 10px;  
}
```

## hasil



## kesimpulan

memberi lengkungan pada border

## Border-style

### penjelasan

Border style adalah properti CSS yang digunakan untuk mengatur gaya garis batas (border) pada elemen HTML.

Nilai yang dapat digunakan dalam properti border-style adalah sebagai berikut:

- `none` : Tidak ada garis batas yang ditampilkan.
- `solid` : Garis batas berupa garis lurus dan terus-menerus.
- `dashed` : Garis batas berupa garis putus-putus.

## program

```
button {  
    border-style:dashed;  
}
```

## hasil



[ **KLIK DISINI** ]

## kesimpulan

Kesimpulannya menggunakan `border-style dashed` akan memberikan garis putus" pada bagian Border button.

## Border color

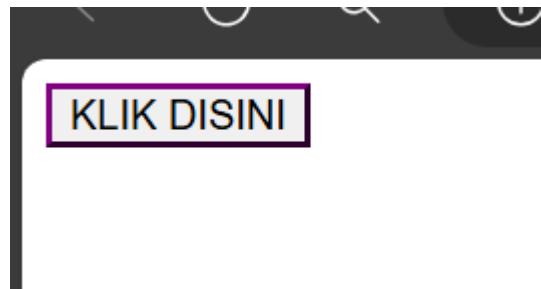
### penjelasan

`Border color` adalah properti CSS yang digunakan untuk mengatur warna garis batas (border) pada elemen HTML.

### program

```
button {  
    border-color:purple;  
}
```

### hasil



### kesimpulan

Memberikan warna pada Bordernya seperti warna purple.

## Padding

### Padding-left

### penjelasan

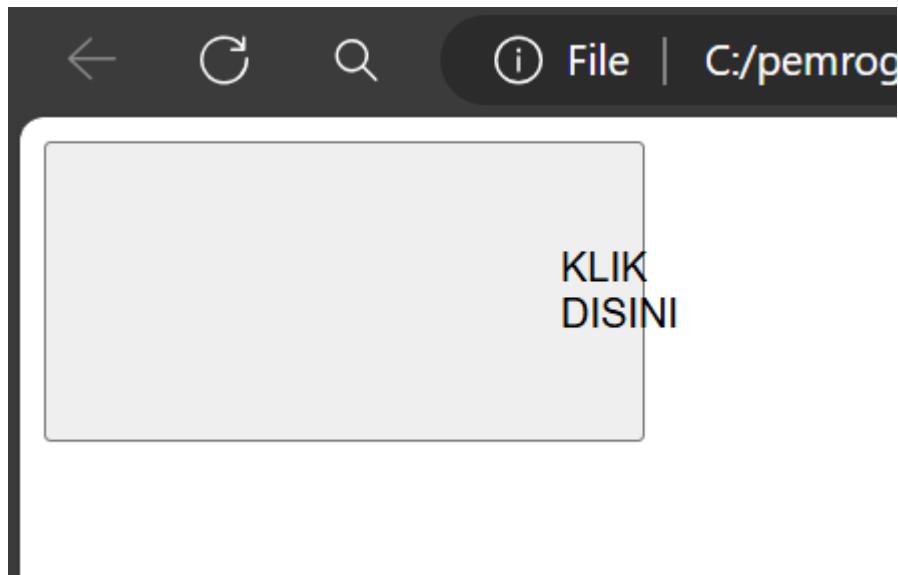
`Padding-left` ini akan membuat bagian kosong di kiri.

### program

```
button {  
    width: 200px ;  
    height: 100px;  
}
```

```
padding-left: 170px;  
}
```

## hasil



## kesimpulan

Membuat bagian kosong dari button yang ada sebelah kiri

## Padding-right

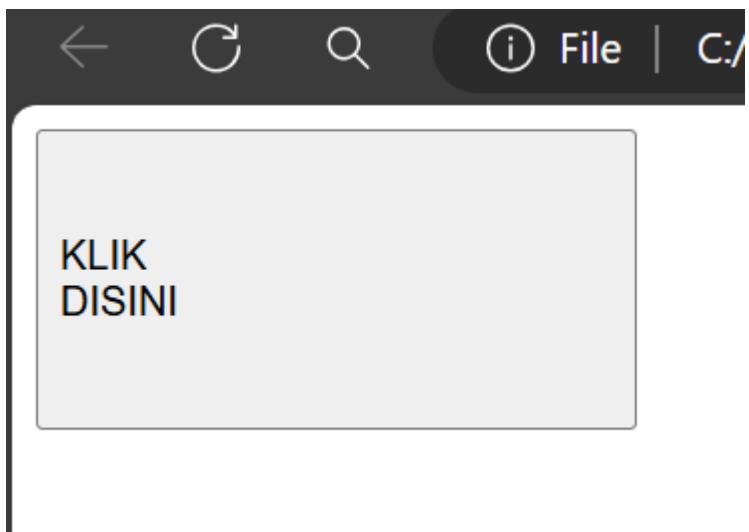
### penjelasan

Padding-right ini akan membuat bagian kosong di kanan.

### program

```
button {  
    width: 200px ;  
    height: 100px;  
    padding-right: 170px;  
}
```

## hasil



## kesimpulan

Membuat bagian kosong dari button yang ada sebelah kanan.

## Padding-bottom

### penjelasan

ini akan membuat tulisan ke atas

## program

```
button {  
    width: 200px ;  
    height: 100px;  
    padding-bottom: 170px;  
}
```

## hasil

KLIK DISINI

## **kesimpulan**

membuat bagian kosong pada bagian bawah

## **Padding-top**

### **penjelasan**

membuat tulisan ke bawah

## **program**

```
button {  
    width: 200px ;  
    height: 100px;  
    padding-top: 150px;  
}
```

## **hasil**



File

C:/pemrograman/index.html

KLIK DISINI

## **kesimpulan**

membuat bagian kosong pada bagian atas

## **Margin**

### **penjelasan**

digunakan untuk memberikan jarak pada button

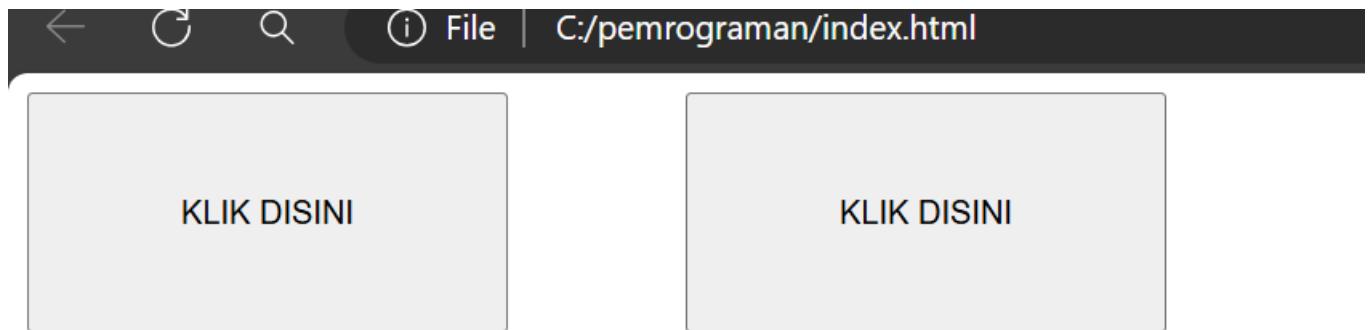
- margin-left memberikan jarak dari kiri
- margin-right memberikan jarak dari kanan
- margin-top memberikan jarak dari atas
- margin-bottom memberikan jarak dari bawah

## **program**

```
.button {  
    width: 200px ;  
    height: 100px;  
    margin-left: 70px;
```

```
}
```

## hasil



## kesimpulan

untuk memberikan jarak antar tepi pada button atau button lainnya.

## Tantangan

### program

### HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title></title>
    <link rel="stylesheet" href="Style.css">
  </head>
  <body>
    <span>
      <p>Selamat Datang<br><b>di Web Zhafran!</b></p>
    </span>
    <button>Klik disini!</button>
  </body>
</html>
```

## CSS

```
body {
    background-color: purple;
}
```

```
img {
    border-radius: 100% 100%;
    border-width: 10px;
    border-style: solid;
    border-color: white;
    margin-left: 220px;
```

```
}
```

```
p {
    font-family: courier;
    font-size: 25px;
    color: black;
    margin-left: ;
    margin-top: -150px;
}
```

```
button {
    width: 100px;
    height: 50px;
    background-color: aqua;
    color: red;
    border-color: red;
    margin-top: ;
    margin-left: 70px;
}
```

**hasil**



File | C:/pemrograman/index.html

Selamat Datang  
di Web Zhafran!

Klik disini!



## Pseudo class

### HOVER

#### Penjelasan

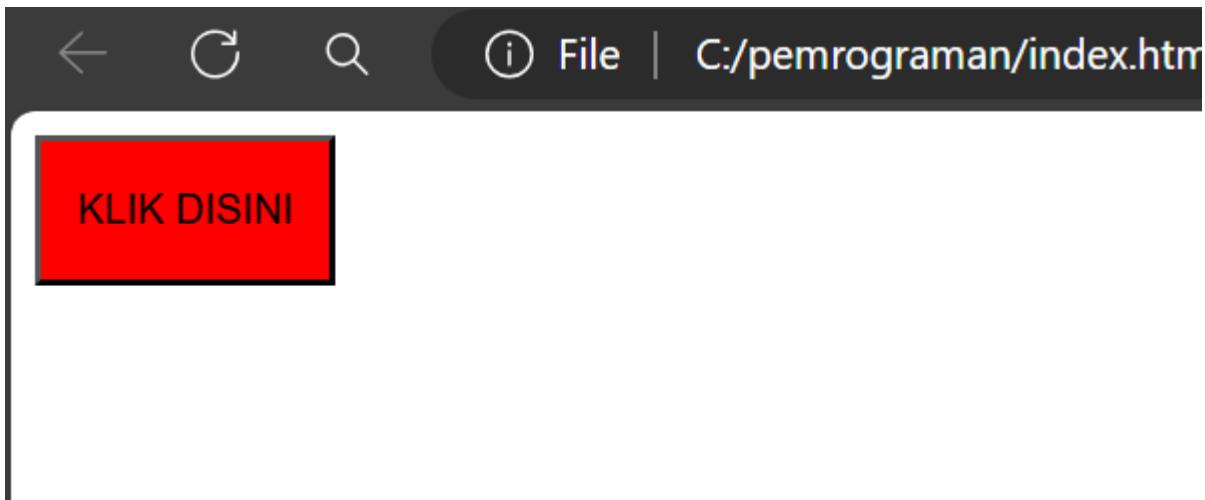
digunakan untuk menentukan gaya elemen ketika pengguna mengarahkan kursor mouse ke elemen tersebut

#### kode program

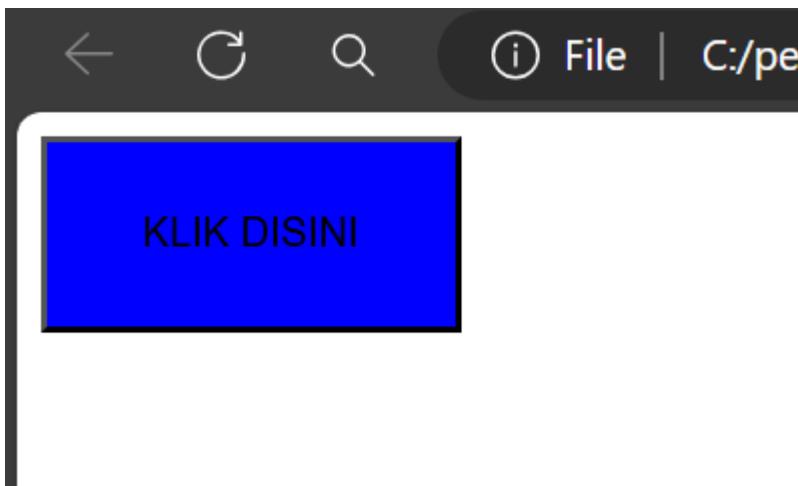
```
button {  
    height: 50px;  
    width: 100px;  
    background-color: red;  
}  
  
button:hover {  
    height: 65px;  
    width: 140px;  
    background-color:blue;  
}
```

**hasil**

**before**



**after**



**kesimpulan**

jika cursor diarahkan ke elemen maka otomatis elemen tersebut berubah bentuk sesuai dengan gaya yang dikasih.

**Active**

**penjelasan**

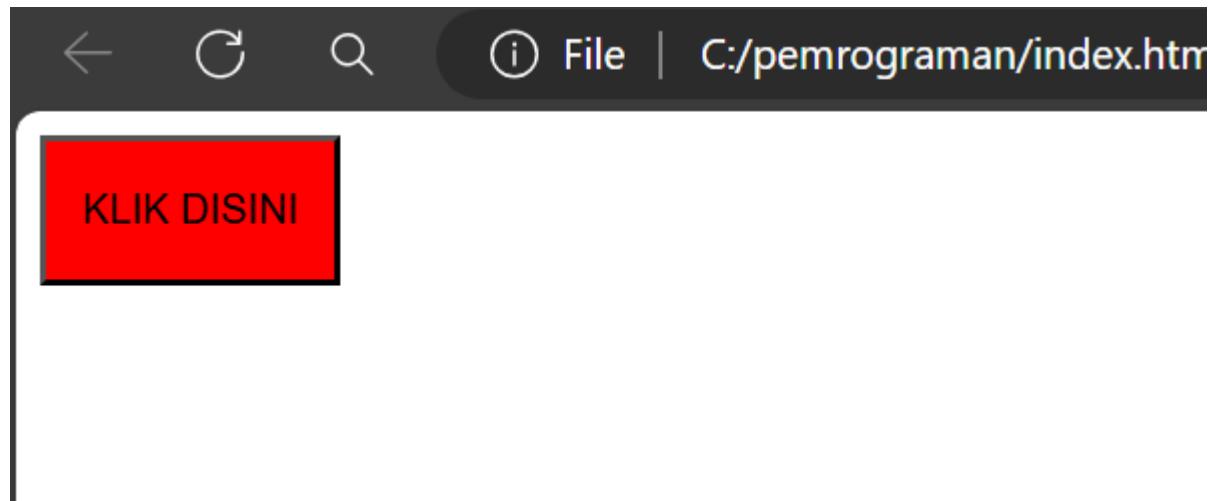
Active digunakan untuk menentukan gaya elemen ketika elemen tersebut aktif/diklik

**program**

```
button {  
    height: 50px;  
    width: 100px;  
    background-color: red;  
}  
  
button:active {  
    height: 65px;  
    width: 140px;  
    background-color:blue;  
}
```

## hasil

### before



### after



## kesimpulan

digunakan jika ingin menambahkan gaya pada elemen yang akan terlihat perubahannya setelah diklik.

## Visited

### penjelasan

`Visited` digunakan untuk menentukan gaya pada elemen yang telah di kunjungi oleh user. umumnya digunakan untuk merubah tampilan tautan yang di klik, jadi user bisa membedakan mana link yang sudah dikunjungi dan mana yang belum dikunjungi.

### program

```
a {  
    color: black;  
}  
  
a:visited {  
    color:blue;  
}
```

### hasil

#### before



KLIK DISINI

#### after



[KLIK DISINI](#)

## kesimpulan

digunakan untuk memberikan gaya pada tampilan link yang sudah dikunjungi.

## link

## penjelasan

Link digunakan untuk memberikan gaya pada elemen/link yang belum dikunjungi, kebalikan dari Visited.

## program

```
a :link {  
font-size: 30px;  
color:black;  
}
```

## hasil

### before

**KLIK DISINI**  
**KLIK**  
**JANGAN DI KLIK**

**after**

**KLIK DISINI**  
**KLIK**  
**JANGAN DI KLIK**

**kesimpulan**

Jadi fungsi dari link ini memberikan gaya pada tautan yang belum dikunjungi.

# Transition

## penjelasan

Transition digunakan untuk mengendalikan perubahan pada sebuah elemen secara halus.

- `Transition-property` menentukan property apa yang akan diberi efek Transition, seperti `width`, `color`, atau `all`(semua property).
- `transition-duration` menentukan waktu berapa lama efek Transitionnya di ukur dalam detik seperti `0,5s`.
- `transition-timing-function` menentukan bagaimana kecepatan perubahan nilai properti selama transisi. Seperti `ease` : lambat diawal, cepat di tengah, dan lambat di akhir `ease-in` : dimulai dengan lambat dan semakin cepat seiring waktu `ease-out` : cepat diawal dan lambat diakhir `ease-in-out` : Kombinasi dari `ease-in` dan `ease-out`, menciptakan efek transisi mulai lambat, cepat di tengah, dan melambat kembali `linear` : memberikan perubahan langsung secara konstan
- `Transition delay` adalah properti dalam CSS yang digunakan untuk menunda mulainya efek transisi pada suatu elemen setelah perubahan nilai properti terjadi.

## program

```
button {  
    height: 50px;  
    width: 100px;  
    background-color: red;  
}  
  
button:hover {  
    height: 60px;  
    width: 110px;  
    background-color: tomato;  
    transition: all 0.5s ease-in-out 0.3s;  
}
```

## hasil

### before



KLIK DISINI!

**after**



KLIK DISINI!

## kesimpulan

Transition digunakan untuk memberikan efek perubahan dalam sebuah elemen.

---

## TANTANGAN Transition

### kode program

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>
```

```
<br>

<span>

Selamat Datang

</span>

<br>

<span>

di <b>Web Zhafran!</b><br>

<button>Klik disini</button>

</span>



</body>

</html>
```

```
body{

background-color: purple;

}

span{

color: white;

font-size: 30px;

font-family: Courier;

}

button{

color: yellow;

background-color: purple;
```

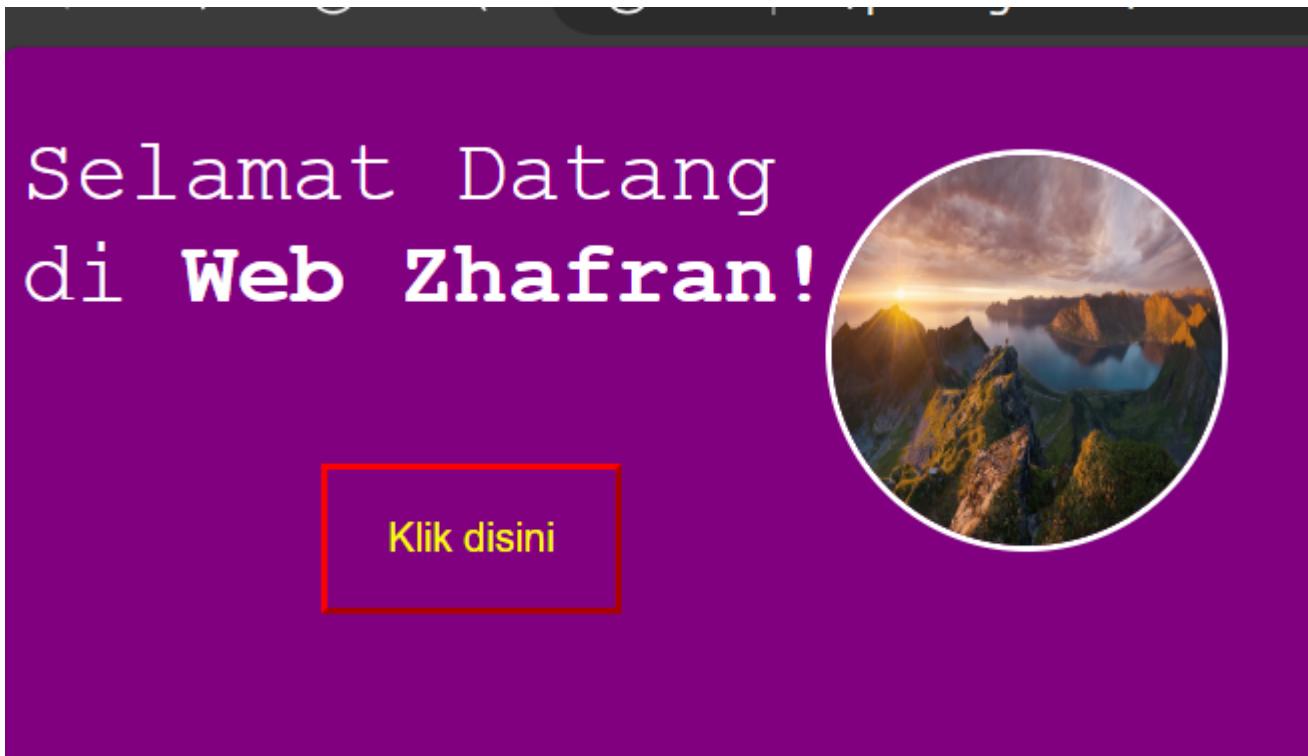
```
border-color: red;  
  
width: 100px;  
  
height: 50px;  
  
margin-left: 100px;  
  
margin-bottom: 1px;  
  
}
```

```
button:hover {  
  
background-color: aqua;  
  
transition: all 0.5s ease-in-out;  
  
color: white;  
  
}
```

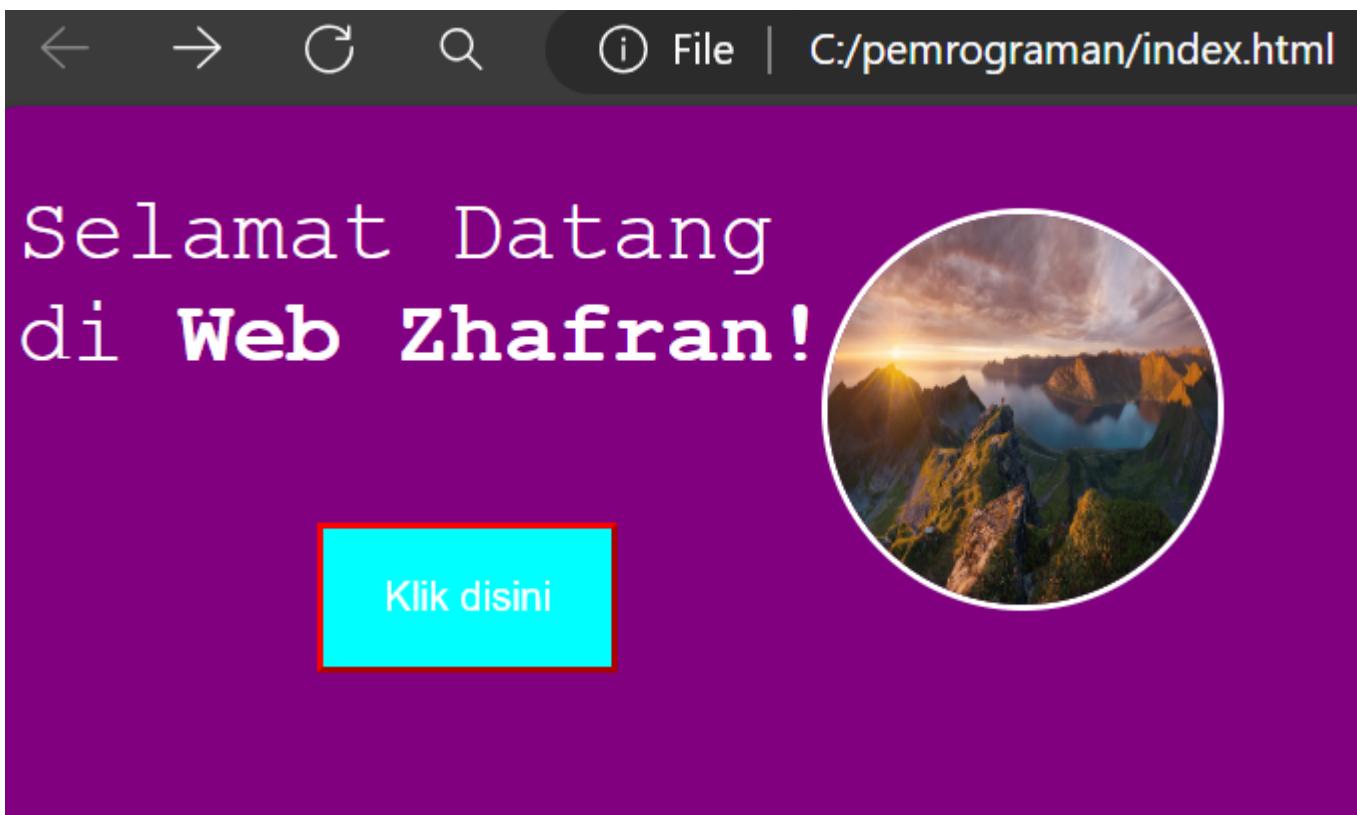
```
img {  
  
width: 130px;  
  
height: 130px;  
  
border-radius: 100px;  
  
border-color: white;  
  
border-style: solid;  
  
margin-left: 50px;  
  
margin-top: -60px;  
  
}
```

hasil

before



after



# Transform

## Scale

### Penjelasan

`scale` adalah bagian dari properti transform yang digunakan untuk mengubah ukuran elemen. Transformasi ini dilakukan tanpa mengubah posisi elemen tersebut dalam tata letak halaman.

### Kode Program

```
<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>CSS</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <button>klik disini</button>
  </body>
</html>
```

```
button {
  height: 40px;
  width: 111px;
  background-color: red;
}
```

```
button:hover {  
    height: 60px;  
    width: 130px;  
    background-color: magenta;  
    transform: scale(0.5);  
}
```

## Hasil

### Before



### After



## Kesimpulan

Transformasi `scale` dapat memperbesar atau memperkecil elemen secara horizontal, vertikal, atau keduanya.

# ScaleX

## Penjelasan

`scaleX` dalam CSS adalah sebuah fungsi transformasi yang digunakan untuk mengubah ukuran elemen secara horizontal. Fungsi ini memungkinkan kita memperbesar atau memperkecil lebar elemen tanpa mengubah tingginya.

## Kode Program

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<button>klik disini</button>

</body>

</html>
```

```
button {

height: 60px;

width: 110px;

background-color: red;

}
```

```
button:hover {  
    height: 70px;  
    width: 120px;  
    background-color: aqua;  
    transform:scaleX(0.5);  
}
```

## Hasil

### Before



### After



## Kesimpulan

`scaleX` dalam CSS adalah alat yang berguna untuk mengubah ukuran elemen secara horizontal

# Rotate

## Penjelasan

`rotate` dalam CSS adalah fungsi transformasi yang digunakan untuk memutar elemen di sekitar titik pusatnya. Fungsi ini memutar elemen yang tegak lurus terhadap layar.

## Kode Program

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<button>klik disini</button>

</body>

</html>
```

```
button {

height: 60px;

width: 110px;

background-color: red;

}
```

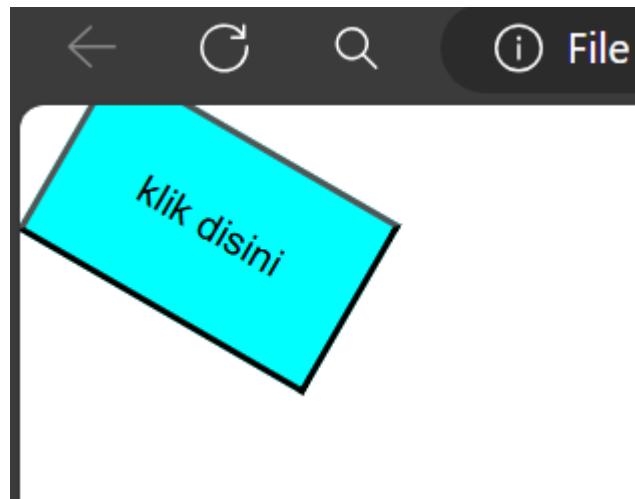
```
button:hover {  
  
    height: 65px;  
  
    width: 110px;  
  
    background-color: aqua;  
  
    transform:rotate(30deg);  
  
}
```

## Hasil

### Before



### After



## Kesimpulan

`rotate` ia akan berputar searah jarum jam jika nilai positif digunakan, dan berlawanan arah jarum jam jika nilai negatif digunakan

---

## SkewX

### Penjelasan

`SkewX` digunakan untuk memiringkan elemen sepanjang horizontal. Nilai `skewX` dinyatakan dengan derajat(deg).

### Kode Program

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<button>klik disini</button>

</body>

</html>
```

```
button {

height: 60px;

width: 110px;

background-color: red;
```

```
}
```

```
button:hover {  
    height: 70px;  
    width: 110px;  
    background-color: aqua;  
    transform: skewX(30deg);  
}
```

## Hasil

### Before



### After



## Kesimpulan

Properti `skewX` mengizinkan kita untuk menentukan sudut miring (`skew angle`) dalam derajat. Ketika kita menggunakan `skewX(angle)`, elemen akan dimiringkan sepanjang sumbu horizontal sebesar sudut yang ditentukan.

---

## Skew

### Penjelasan

`Skew` digunakan untuk memiringkan elemen baik secara horizontal maupun vertikal dalam satu nilai.

### Kode Program

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<button>klik disini</button>

</body>

</html>
```

```
button {

height: 60px;

width: 110px;

background-color: red;
```

```
}
```

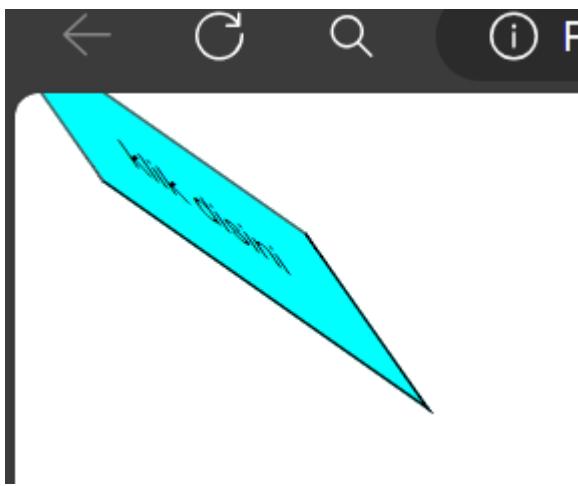
```
button:hover {  
    height: 60px;  
    width: 110px;  
    background-color: aqua;  
    transform: skew(35deg, 35deg);  
}
```

## Hasil

### Before



### After



## Kesimpulan

Properti `skew` memungkinkan kita untuk menentukan dua sudut miring, satu untuk sumbu X dan satu untuk sumbu Y, dalam derajat.

---

## Translate

### Penjelasan

`Translate` digunakan untuk memindahkan elemen dari posisi awalnya. ketika pengguna mengarahkan kursor ke elemen button, elemen tersebut akan digeser 85 piksel ke kanan dan 40 piksel ke bawah.

### Kode Program

```
<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>CSS</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
```

```
<button>klik disini</button>  
</body>  
</html>
```

```
button {  
  
height: 60px;  
  
width: 110px;  
  
background-color: red;  
}
```

```
button:hover {  
  
height: 60px;  
  
width: 110px;  
  
background-color: aqua;  
  
transform:translate(85px, 40px);  
}
```

## Hasil

### Before



## After



## Kesimpulan

Properti `translate` memungkinkan kita untuk menentukan pergeseran (offset) dalam piksel atau persentase terhadap ukuran elemen terkait.

---

## Matrix

### Penjelasan

`Matrix` adalah salah satu fungsi transformasi yang digunakan untuk mengubah tata letak elemen HTML.

Contoh property value nya seperti `matrix(1, -0.3, 0.6, 1, 50, 20)`.

### Kode Program

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<button>klik disini</button>

</body>

</html>
```

```
button {

height: 60px;

width: 110px;

background-color: red;

}

button:hover {

height: 60px;

width: 110px;

background-color: aqua;

transform: matrix(1, -0.3, 0.6, 1, 50, 20);
```

}

## Hasil

### Before



### After



## Kesimpulan

Penggunaan matrix memungkinkan kita untuk mengontrol transformasi dengan tingkat kebebasan yang tinggi, tetapi sintaksnya memerlukan pemahaman tentang operasi matriks dan pengaruhnya terhadap elemen yang ditransformasi.

---

## FlexBox

# FLEX CONTAINER

## Display Flex

### Penjelasan

`display: flex;` adalah sebuah properti CSS yang digunakan untuk mengatur sebuah container agar anak-anak elemennya (child elements) menjadi flex items dan mengaktifkan model layout Flexbox. Ini memberikan kontrol yang sangat fleksibel dalam pengaturan tata letak elemen di dalam container, baik secara horizontal maupun vertikal.

### Kode Program

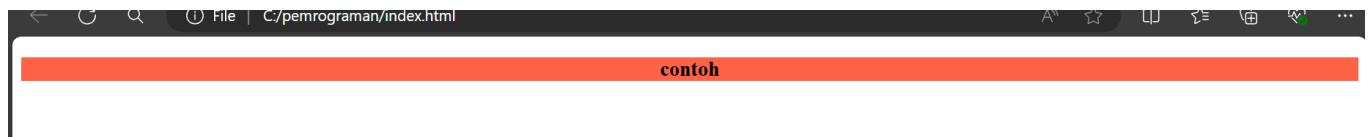
#### HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>flexbox</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <h3>contoh</h3>
  </body>
</html>
```

#### CSS

```
h3 {
  display: flex;
  justify-content: center;
  background-color: crimson;
}
```

### Hasil



### Kesimpulan

`display: flex;` mengaktifkan model layout Flexbox di sebuah container CSS, yang memungkinkan pengaturan tata letak yang fleksibel untuk elemen-elemen di dalamnya.

---

## FLEX DIRECTION

### Penjelasan

`Flex direction` digunakan untuk mengatur arah tata letak elemen dalam sebuah flexbox container.

Beberapa contoh property value nya :

- `row` : Flex item disusun dalam satu baris dari kiri ke kanan.
- `row-reverse` : Sama seperti `row`, tetapi urutan item dibalik, dari kanan ke kiri.
- `column` : Flex item disusun dalam satu kolom dari atas ke bawah.
- `column-reverse` : Sama seperti `column`, tetapi urutan item dibalik, dari bawah ke atas

### Kode Program

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<div class="container">

<div class="item">Box-1</div>

<div class="item">Box-2</div>

<div class="item">Box-3</div>
```

```
<div class="item">Box-4</div>

</div>

</body>

</html>
```

```
.container {

  display: flex;

  flex-direction: column;

  border: 2px solid black;

  height: 200px;

  width: 400px;

}
```

```
.item {

  background-color: red;

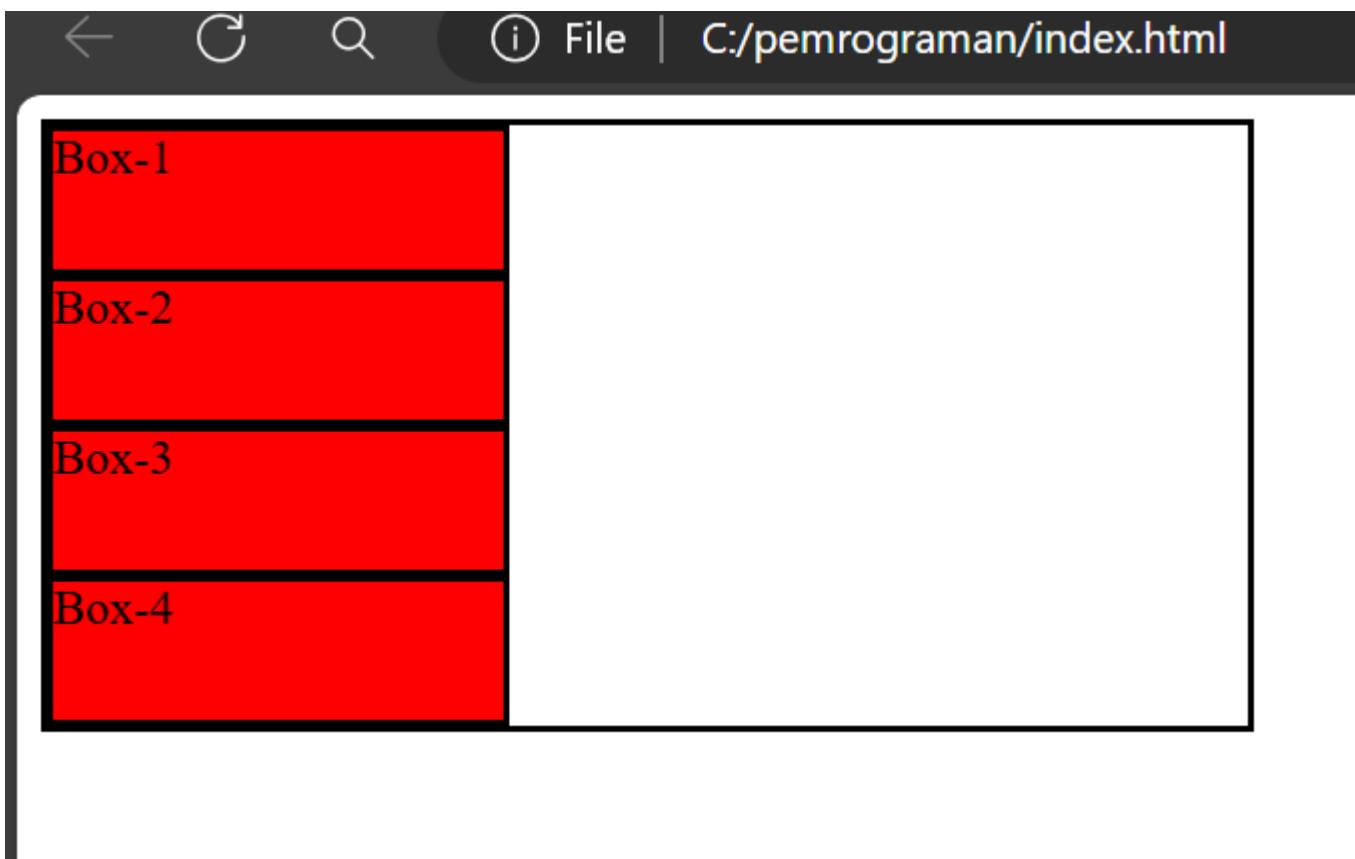
  border: 2px solid black;

  height: 50px;

  width: 150px;

}
```

## Hasil



## Kesimpulan

`Flex direction` berfungsi ketika kita ingin mengatur tata letak item yang berada didalam container.

---

## FLEX WRAP

### Penjelasan

`Flex wrap` dalam flexbox untuk mengontrol apakah item-item flexbox dalam sebuah container akan melintasi baris atau tidak ketika ruang yang tersedia tidak cukup untuk menampung semua item tersebut dalam satu baris atau kolom.

Beberapa Property Value nya :

- `wrap` berfungsi ketika dalam 1 baris tidak dapat lagi di tampung maka akan membuat baris baru di bawahnya
- `nowrap` item-item akan berada dalam satu baris atau kolom, meskipun ruang tidak cukup.
- `wrap-reverse` sama dengan `wrap` cuman kebalikannya saja.

### Kode Program

```
<html>
  <head>
    <title>flexbox</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="container">
      <div class="item">Box-1</div>
      <div class="item">Box-2</div>
      <div class="item">Box-3</div>
      <div class="item">Box-4</div>
    </div>
  </body>
</html>
```

```
.container {

  display: flex;

  flex-wrap: wrap;

  border: 2px solid black;

  height: 200px;

  width: 400px;

}
```

```
.item {

  background-color: red;

  border: 2px solid black;

  height: 50px;

  width: 150px;

}
```

## Hasil



## Kesimpulan

`Flex wrap` dominan digunakan ketika item yang berada dalam 1 baris sudah tidak cukup di tampung oleh container, maka item selanjutnya akan membuat baris baru.

---

## ALIGN ITEMS

### Penjelasan

`Align items` Property CSS yang digunakan untuk mengatur tataletak elemen secara vertikal dalam flex container.

Beberapa contoh Property Value nya :

- `Flex-start` item akan diletakkan di awal container
- `Flex-end` item akan diletakkan di akhir container
- `center` item akan diletakkan di tengah container
- `stretch` item akan meregang mengisi seluruh tinggi container
- `base-line` item akan diletakkan digaris dasar dari Text mereka.

## Kode Program

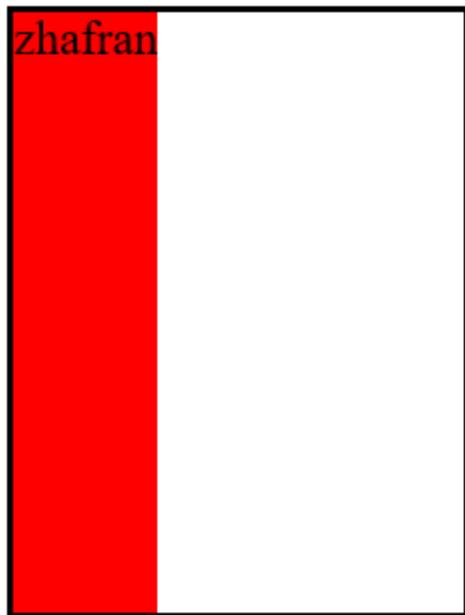
```
<html>
  <head>
    <title>flexbox</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="container">
      <div class="item">Taufiq</div>
    </div>

  </body>
</html>
```

```
.container {
  display: flex;
  height: 200px;
  width: 150px;
  border: 2px solid black;
  align-items: stretch;
}
```

```
.item {
  background-color: red;
}
```

## Hasil



## Kesimpulan

Align items digunakan untuk mengatur tataletak suatu elemen secara vertikal

---

## JUSTIFY CONTENT

### Penjelasan

justify-content adalah properti CSS yang digunakan untuk mengatur cara penempatan dan penyebaran ruang ekstra dari sebuah flex container secara horizontal.

Beberapa contoh property value nya :

- flex-start : Item-item akan diletakkan di awal container.
- flex-end : Item-item akan diletakkan di akhir container.
- center : Item-item akan diletakkan di tengah container.
- space-between : Item-item akan ditempatkan dengan jarak yang sama di antara mereka, tetapi tidak di sisi kanan dan kiri container.
- space-around : Item-item akan ditempatkan dengan jarak yang sama di antara mereka, termasuk di sisi kanan dan kiri container.

- `space-evenly` : Item-item akan ditempatkan dengan jarak yang sama di antara mereka, termasuk di sisi kanan dan kiri container, serta jarak yang sama di sekeliling mereka.

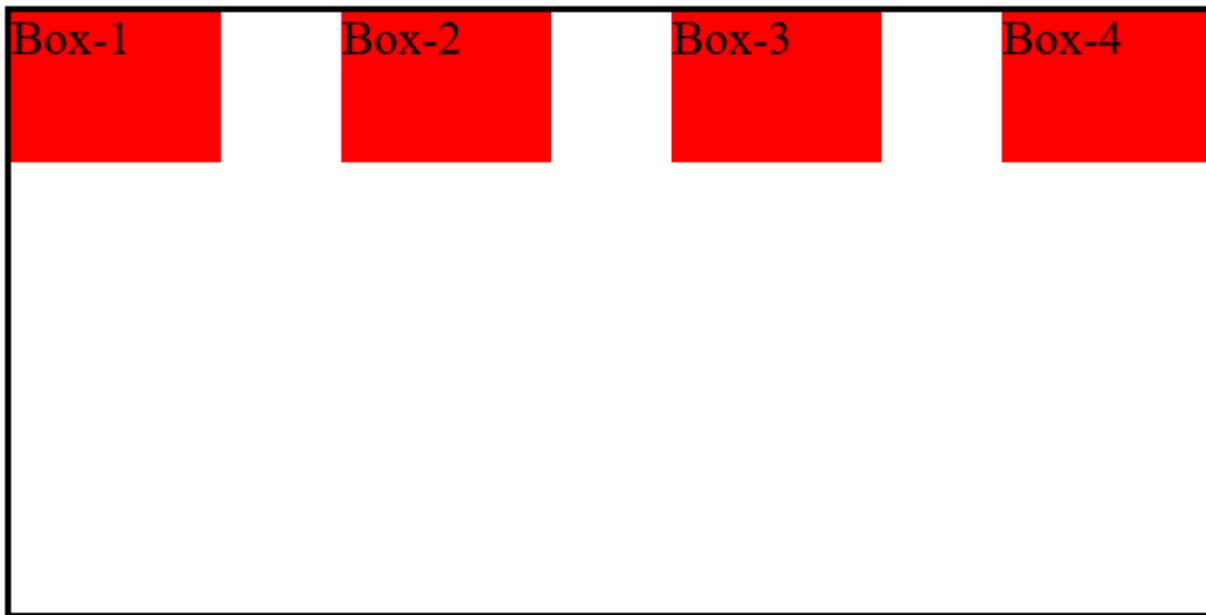
## Kode Program

```
<html>
  <head>
    <title>flexbox</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="container">
      <div class="item">Box-1</div>
      <div class="item">Box-2</div>
      <div class="item">Box-3</div>
      <div class="item">Box-4</div>
    </div>
  </body>
</html>
```

```
.container {
  display: flex;
  height: 200px;
  width: 400px;
  border: 2px solid black;
  justify-content: space-between;
}

.item {
  background-color: red;
  height: 50px;
  width: 70px;
}
```

## Hasil



## Kesimpulan

`Justify content` digunakan untuk mengatur jarak antar elemen secara horizontal atau baris

---

## ALIGN CONTENT

### Penjelasan

`Align-content` adalah properti CSS yang digunakan untuk mengatur cara konten dalam sebuah flex container disusun secara vertikal ketika ada ruang ekstra di dalam kontainer flex tersebut.

Beberapa Property Value nya :

- `flex-start` : item akan berada di awal container
- `flex-end` : item akan berada di akhir container
- `center` : item akan diletakkan ditengah container
- `space-between` : item diperintahkan diletakkan dengan jarak yang sama, pertama di awal dan terakhir di akhir.

- `space-around` : item diperintahkan diletakkan dengan jarak yang sama di sekitar setiap item.
- `space-evenly` : item diperintahkan untuk diletakkan dengan jarak yang sama di antara dan sekitar setiap item.
- `stretch` : item diperintahkan untuk meregangkan untuk mengisi seluruh kontainer fleksibel.

## Kode Program

```
<html>
  <head>
    <title>flexbox</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="container">
      <div class="item">Box-1</div>
      <div class="item">Box-2</div>
      <div class="item">Box-3</div>
      <div class="item">Box-4</div>
    </div>

  </body>
</html>
```

```
.container {
  display: flex;
  height: 500px;
  width: 400px;
  border: 2px solid black;
  align-content: space-around;
  flex-wrap: wrap;
}

.item {
  background-color: red;
  height: 50px;
  width: 150px;
}
```

## Hasil

Box-1

Box-2

Box-3

Box-4

## Kesimpulan

Align content digunakan untuk mengatur jarak antar elemen secara vertikal

---

## FLEX ITEM

## ORDER

## Penjelasan

Order adalah Property CSS dalam flexbox yang digunakan untuk mengatur tata letak flex item dalam sebuah flex container, value yang di berikan kepada order adalah bilangan bulat.

## Kode Program

```
<html>
  <head>
    <title>flexbox</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="container">
      <div class="item satu">Box-1</div>
      <div class="item dua">Box-2</div>
      <div class="item tiga">Box-3</div>
      <div class="item empat">Box-4</div>
    </div>
  </body>
</html>
```

```
.container {
  display: flex;
  flex-direction: row;
  border: 2px solid yellow;
  height: auto;
  width: 400px;
}

.item {
  background-color: red;
  border: 2px solid black;
  height: 50px;
  width: 150px;
  margin: 5px;
}

.satu {
  order: 4;
}
```

## Hasil



## Kesimpulan

Order digunakan ketika kita ingin mengatur tataletak flex item dalam flex container

---

## FLEX-GROW

### Penjelasan

Flex-Grow adalah Property CSS yang mengatur seberapa besar flex item akan memperluas dirinya dalam flex container relatif terhadap item-item lain di dalam container tersebut. Nilai dari `flex-grow` menentukan seberapa banyak ruang tambahan yang akan diambil oleh item flex dalam container jika ada ruang kosong yang tersedia setelah item-item lain telah menempati ruang mereka sendiri.

### Kode Program

```
<html>
  <head>
    <title>flexbox</title>
    <link rel="stylesheet" href="style.css">
  </head>

  <body>
    <div class="container">
      <div class="item satu">Box-1</div>
      <div class="item dua">Box-2</div>
      <div class="item tiga">Box-3</div>
      <div class="item empat">Box-4</div>
    </div>
```

```
</body>  
</html>
```

```
.container {  
  
    display: flex;  
  
    flex-direction: row;  
  
    border: 2px solid yellow;  
  
    height: auto;  
  
    width: 400px;  
  
}
```

```
.item {  
  
    display: flex;  
  
    background-color: red;  
  
    border: 2px solid black;  
  
    height: 50px;  
  
    margin: 5px;  
  
}
```

```
.satu {  
  
    flex-grow: 2;  
  
}
```

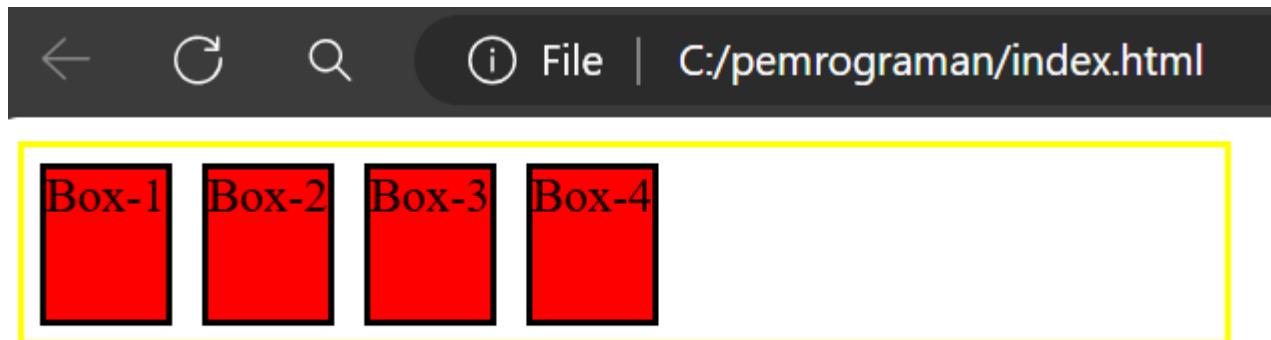
```
.dua {
```

```
flex-grow: 1;  
}
```

```
.tiga {  
  
flex-grow: 1;  
}
```

```
.empat {  
  
flex-grow: 1;  
}
```

## Hasil



## Kesimpulan

Flex-Grow digunakan ketika kita ingin memperluas salah satu flex item dalam sebuah flex container.

---

## FLEX-SHRINK

### Penjelasan

`flex-shrink` adalah properti dalam CSS yang digunakan dalam konteks Flexbox untuk mengontrol seberapa besar sebuah elemen fleksibel dapat menyusut jika tidak ada cukup ruang di dalam kontainernya. Properti ini mengatur tingkat penyusutan relatif dari elemen-elemen fleksibel dalam sebuah flex container.

## Kode Program

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Contoh Penggunaan flex-shrink</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>

<div class="flex-container">
  <div class="flex-item item1">Elemen 1</div>
  <div class="flex-item item2">Elemen 2</div>
  <div class="flex-item item3">Elemen 3</div>
</div>

</body>
</html>
```

```
.flex-container {

  display: flex;
}

.flex-item {

  flex: 1;

  padding: 10px;

  border: 1px solid black;
}
```

```
.item1 {  
  flex-shrink: 1;  
}  
  
.item2, .item3 {  
  flex-shrink: 2;  
}
```

## Hasil

Elemen 1	Elemen 2	Elemen 3

## Kesimpulan

Flex-shrink mengatur ukuran item ketika ruang tidak cukup.

---

## FLEX-BASIS

### Penjelasan

flex-basis adalah properti CSS dalam desain tata letak flexbox yang menentukan ukuran awal (basis) dari item fleksibel sebelum penyesuaian ukuran fleksibel terjadi.

### Kode Program

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
<meta charset="UTF-8">  
<meta name="viewport" content="width=device-width, initial-scale=1.0">  
<title>Contoh Penggunaan flex-basis</title>
```

```
<link rel="stylesheet" href="style.css">
</head>
<body>

<div class="flex-container">
  <div class="flex-item item1">Elemen 1</div>
  <div class="flex-item item2">Elemen 2</div>
  <div class="flex-item item3">Elemen 3</div>
</div>

</body>
</html>
```

```
.flex-container {
```

```
  display: flex;
```

```
}
```

```
.flex-item {
```

```
  flex-grow: 1;
```

```
  border: 1px solid black;
```

```
  margin: 5px;
```

```
}
```

```
.item1 {
```

```
  flex-basis: 50px;
```

```
}
```

```
.item2 {
```

```
  flex-basis: 70px;
```

```
}
```

```
.item3 {  
    flex-basis: 90px;  
}
```

## Hasil



## Kesimpulan

`flex-basis` menentukan ukuran awal elemen sebelum fleksbox membagikan ruang yang tersisa.

---

## ALIGN-SELF

### Penjelasan

`align-self` adalah properti CSS yang digunakan dalam konteks Flexbox untuk mengontrol penempatan vertikal individu dari elemen flex dalam container. Properti ini mengatur penempatan vertikal elemen tunggal dalam halaman, mengesampingkan nilai `align-items` yang diterapkan pada container

Nilai yang umum digunakan untuk `align-self` adalah:

- `flex-start` : Elemen diletakkan di bagian atas container.
- `flex-end` : Elemen diletakkan di bagian bawah container.
- `center` : Elemen diletakkan di tengah container.
- `baseline` : Elemen diletakkan pada garis dasar container.
- `stretch` : Elemen diregangkan untuk mencakup tinggi container.

### Kode Program

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Contoh Penggunaan align-self</title>
<link rel="stylesheet" href="style.css">
<style>
    .flex-container {
        display: flex;
        height: 200px; /* Tinggi kontainer */
        border: 1px solid black;
    }

    .flex-item {
        width: 100px;
        margin: 5px;
        background-color: lightblue;
    }

    .item1 {
        align-self: flex-start;
    }

    .item2 {
        align-self: center;
    }

    .item3 {
        align-self: flex-end;
    }
</style>
</head>
<body>
<div class="flex-container">
    <div class="flex-item item1">Elemen 1</div>
    <div class="flex-item item2">Elemen 2</div>
    <div class="flex-item item3">Elemen 3</div>
</div>
</body>
</html>
```

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

```
height: 200px;
border: 1px solid black;
}

.flex-item {
  width: 100px;
  margin: 5px;
  background-color: red;
}

.item1 {
  align-self: flex-start;
}

.item2 {
  align-self: center;
}

.item3 {
  align-self: flex-end;
}
```

## Hasil

Elemen 1

Elemen 2

Elemen 3

## Kesimpulan

Align self mengatur penempatan item secara vertikal

# FLEX

## Penjelasan

Properti `flex` memungkinkan Anda untuk secara singkat menentukan bagaimana elemen flex akan mengisi ruang dalam flex container. Dengan menggunakan `flex`, Anda dapat mengatur elemen flex untuk: `flex-grow`, `flex-shrink`, dan `flex-basis`.

## Kode Program

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Contoh Penggunaan flex</title>
<link rel="stylesheet" href="style.css">
</head>
<body>

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

</body>
</html>
```

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

.flex-item {
    flex: 1;
    border: 1px solid black;
    margin: 5px;
}
```

## Hasil

Elemen 1	Elemen 2	Elemen 3
----------	----------	----------

## Kesimpulan

Flex merupakan gabungan dari flex grow, flex shrink, flex basis

# TANTANGAN FLEXBOX

## Kode Program

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<div class="container">

<div class="box-1">

<p class="desk">Selamat datang<br>di<span> Web Zhafran</span></p>

<button class="btn">Klik disini</button>

</div>



</div>
```

```
</body>
```

```
</html>
```

```
.container {  
  
    display: flex;  
  
    flex-direction: row;  
  
    background-color: purple;  
  
    width: 800px;  
  
    height: 400px;  
  
}
```

```
.box-1 {  
  
    display: flex;  
  
    flex-direction: column;  
  
    align-items: end;  
  
    margin-right: 50px;  
  
    margin-left: 10px;  
  
    margin-top: 15px;  
  
}
```

```
.desk {  
  
    padding-top: 5px;  
  
    font-size: 50px;  
  
    font-family: 'Courier New', Courier, monospace;
```

```
}

span {
    font-weight: bold;
}

.btn {
    height: 80px;
    width: 120px;
    border: 3px solid black;
}

.btn:hover {
    transform: translate(-170px, -60px);
    transition: all 0.5s ease-in-out 0.3s;
    border: 3px solid yellow;
}

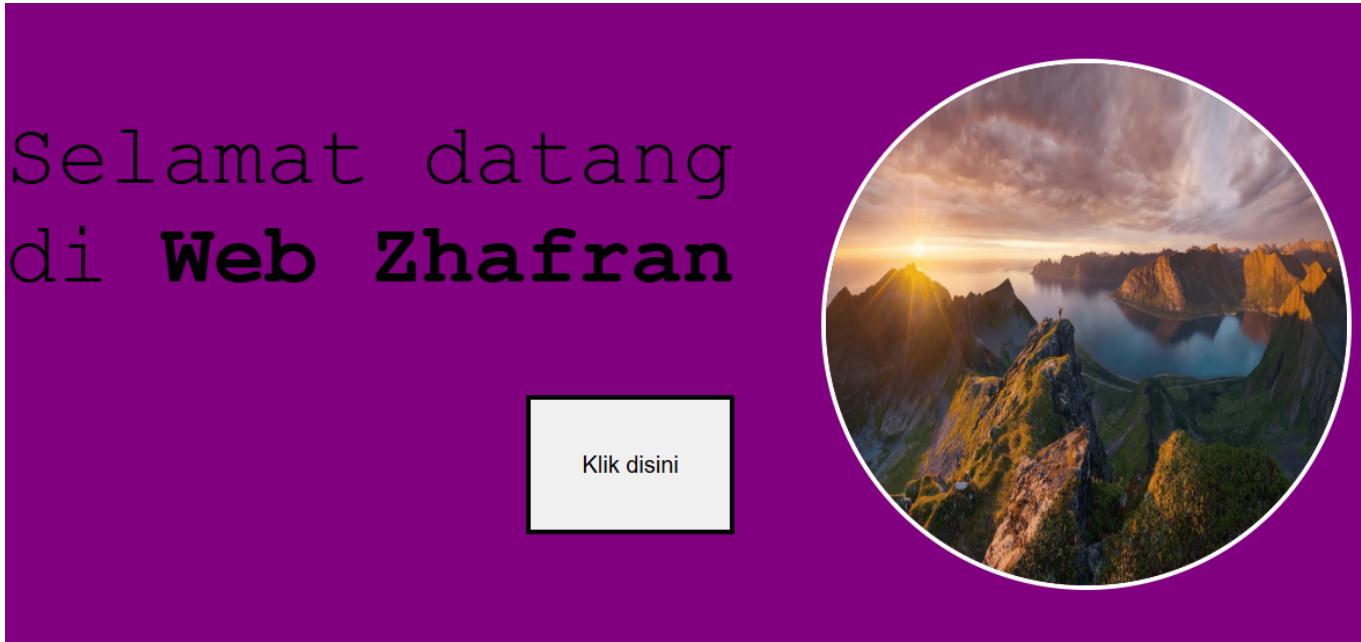
img {
    height: 300px;
    width: 300px;
    border-radius: 100%;
    border: 3px solid white;
```

```
margin-top: 40px;
```

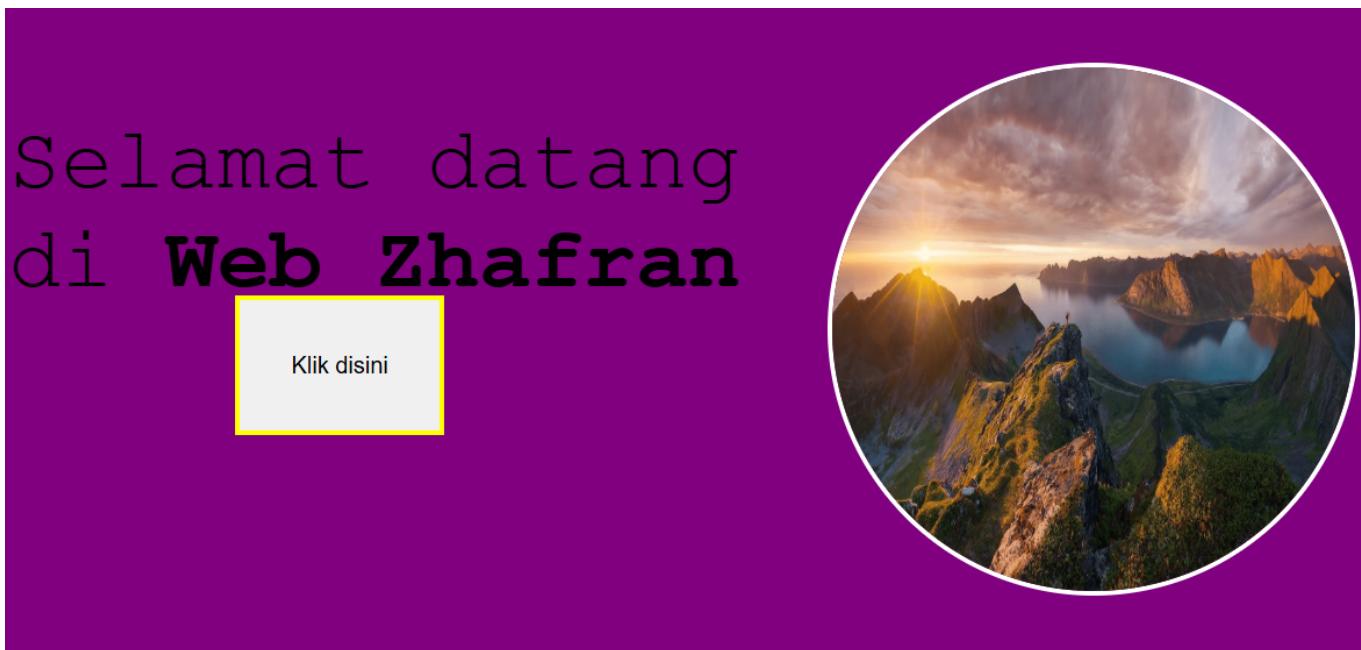
```
}
```

## Hasil

before



after



# Position

## Position Relative

### Penjelasan

Position Relative adalah posisi mirip seperti static position, dimana element akan ditempatkan sesuai normal flow, ketika elemen tersebut mau di pindahkan dia akan pindah dari posisi awalnya.

### Kode Program

```
<html>
  <head>
    <title>position</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="container">
      <div class="item1"></div>
      <div class="item2"></div>
    </div>
  </body>
</html>
```

```
.container {

  display: flex;

  flex-direction: row;

  width: 150px;

  height: 150px;

  background-color: red;

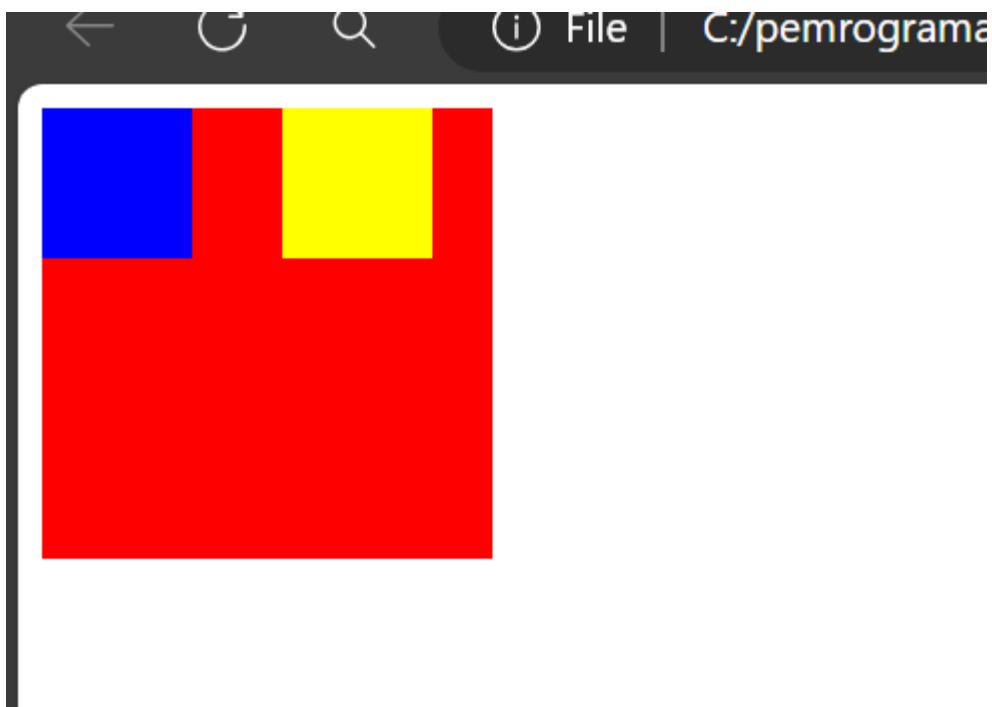
}
```

```
.item1 {

  width: 50px;
```

```
height: 50px;  
  
background-color: blue;  
  
}  
  
  
.item2 {  
  
width: 50px;  
  
height: 50px;  
  
background-color: yellow;  
  
position: relative;  
  
left: 30px;  
  
}
```

## Hasil



## Kesimpulan

Position relative berpindah dari tempat awalnya yang akan mengganggu posisi elemen lain.

# Position Absolute

## Penjelasan

Posisi `absolute` dalam CSS mengarahkan sebuah elemen untuk ditempatkan secara independen di dalam elemen tertentu, baik itu elemen yang berposisi relatif atau seluruh halaman (viewport) jika tidak ada elemen yang berposisi relatif.

## Kode Program

```
<html>
  <head>
    <title>position</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="container">
      <div class="item1"></div>
      <div class="item2"></div>
    </div>
  </body>
</html>
```

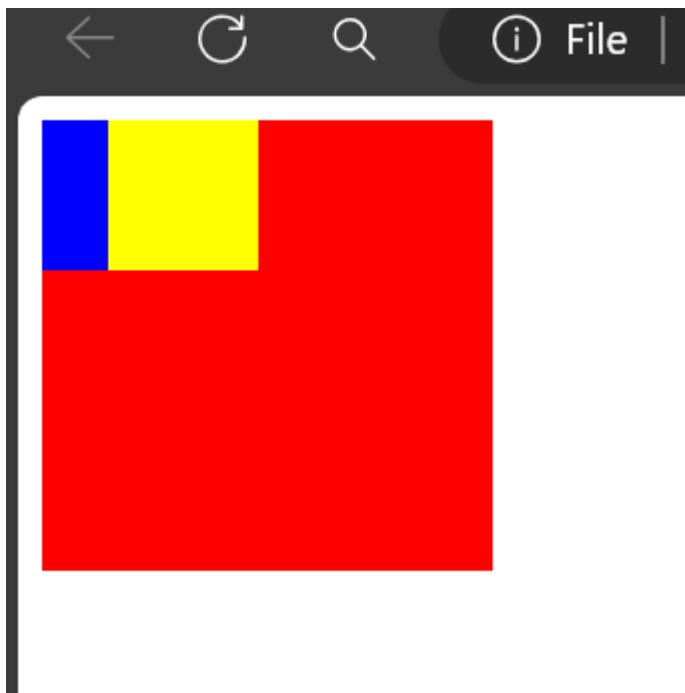
```
.container {
  display: flex;
  flex-direction: row;
  width: 150px;
  height: 150px;
  background-color: red;
}
```

```
.item1 {
  width: 50px;
```

```
height: 50px;  
  
background-color: blue;  
  
}  
  
}
```

```
.item2 {  
  
width: 50px;  
  
height: 50px;  
  
background-color: yellow;  
  
position: absolute;  
  
left: 30px;  
  
}
```

## Hasil



## Kesimpulan

`Position absolute` dia berpindah mengikuti parentnya, tanpa mempengaruhi tata letak elemen-elemen lain.

---

## Position Fixed

### Penjelasan

`Position Fixed` ketika suatu elemen memiliki properti position yang di atur menjadi fixed maka elemen Tersebut akan terus berada dalam viewport yang sudah di tentukan, bahkan saat halaman di gulir.

### Kode Program

```
<html>
  <head>
    <title>position</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="container">FIXED</div>
    <div class="item"></div>
    <div class="item2"></div>
  </body>
</html>
```

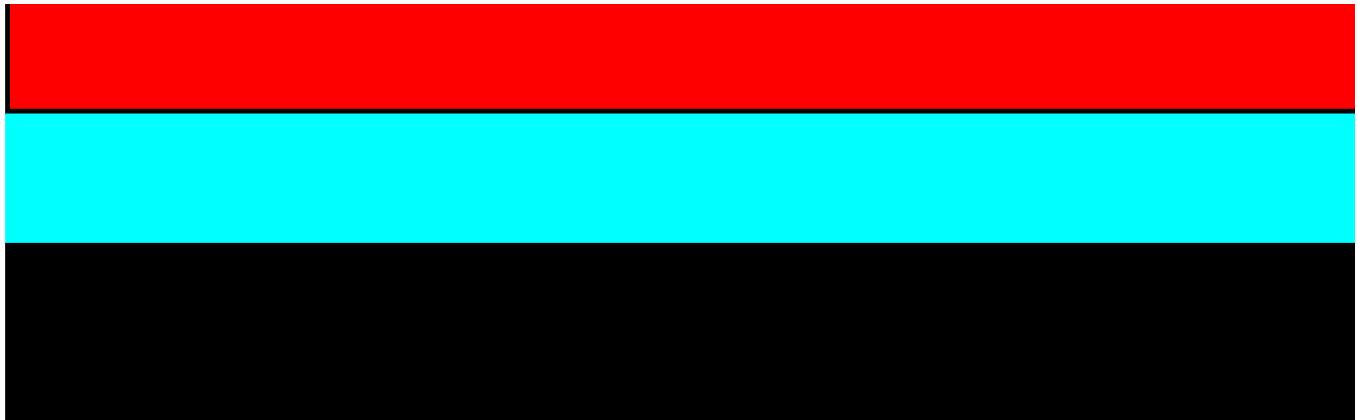
```
.container {
  width: 100vw;
  height: 150px;
  background-color: red;
  border: 4px solid black;
  position: fixed;
}
```

```
.item {
```

```
width:100vw;  
  
height:300px;  
  
background-color: aqua;  
  
}
```

```
.item2 {  
  
width:100vw;  
  
height:300px;  
  
background-color: black;  
  
}
```

## Hasil



## Kesimpulan

Position Fixed digunakan ketika ada elemen, kita ingin dia berada dalam viewport terus.

---

## Position Sticky

### Penjelasan

Position Sticky gabungan dari `relative` dan `Fixed`, elemen yang di berikan position Sticky akan berperilaku relative sampe dengan titik yang ditentukan baru berubah menjadi fixed.

## Kode Program

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<h2>PERCOBAAN1</h2>

<h2>PERCOBAAN2</h2>

<h2 class="sticy">CONTOH</h2>

<h2>PERCOBAAN3</h2>

</body>

</body>

</html>
```

```
.sticy {

background-color: aqua;

width: 100vw;
```

```
position:sticky;  
  
top: 50px;  
  
}
```

## Hasil

**PERCOBAAN1**

**PERCOBAAN2**

**CONTOH**

**PERCOBAAN3**

## Kesimpulan

Sticky digunakan ketika ada suatu elemen yang terdapat dalam baris, akan tetapi kita mau dia berada dalam viewport terus menerus tanpa harus ikut ke scroll.

---

## TANTANGAN POSITION

### Kode Program

```
<!DOCTYPE html>  
  
<html lang="en">  
  
<head>  
  
<meta charset="UTF-8">
```

```
<title>CSS</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<body>

<div class="container">

<div class="item1">

</div>

<div class="item2">

    <p class="tanggal">Thursday, March 28, 2024</p>

    <p class="judul">The standard chunk of Lorem Ipsum</p>

    <p class="catatan">Sed poseure consecetur est at lobortis. aenean eu leo quam.</p>

</div>

<div class="item3">

    <p class="text">Read More</p>

</div>

</div>
```

```
</body>
```

```
</body>
```

```
</html>
```

```
body{
```

```
  display: flex;
```

```
  justify-content: center;
```

```
  background-color: blue;
```

```
}
```

```
.container {
```

```
  display: flex;
```

```
  flex-direction: column;
```

```
  height: 500px;
```

```
  width: 350px;
```

```
  margin-top: 50px;
```

```
}
```

```
.item1 {
```

```
  height: 250px;
```

```
}
```

```
.item2 {
```

```
background-color:white;  
height: 190px;  
padding-left: 20px;  
}
```

```
.gambar {  
border-radius: 50%;  
position: relative;  
left: 270px;  
top: -17px;  
}
```

```
.tanggal {  
font-size: 15px;  
color: rgb(109, 111, 112);  
margin-top: -25px;  
}
```

```
.judul {  
font-size: 30px;  
font-weight: bold;  
margin-top: -5px;
```

```
}
```

```
.catatan {  
    font-size: 15px;  
    margin-top: -15px;  
}
```

```
.item3 {  
    display: flex;  
    flex-direction: row;  
    justify-content: space-between;  
    align-items: center;  
    background-color:rgb(160, 170, 160);  
    height: 50px;  
}
```

```
.text {  
    padding-left: 20px;  
    font-size: 18px;  
}
```

```
.icon {  
    border-radius: 80%;
```

```
padding-right: 10px;  
}  
  
Hasil
```



Thursday, March 28,2024

# The standard chunk of Lorem Ipsum

Sed poseure consectetur est at lobortis. aenean eu leo quam.

Read More



