

CSS Esensial

Modul Ekstrakurikuler Coding — SMPIT Nur Hidayah Surakarta

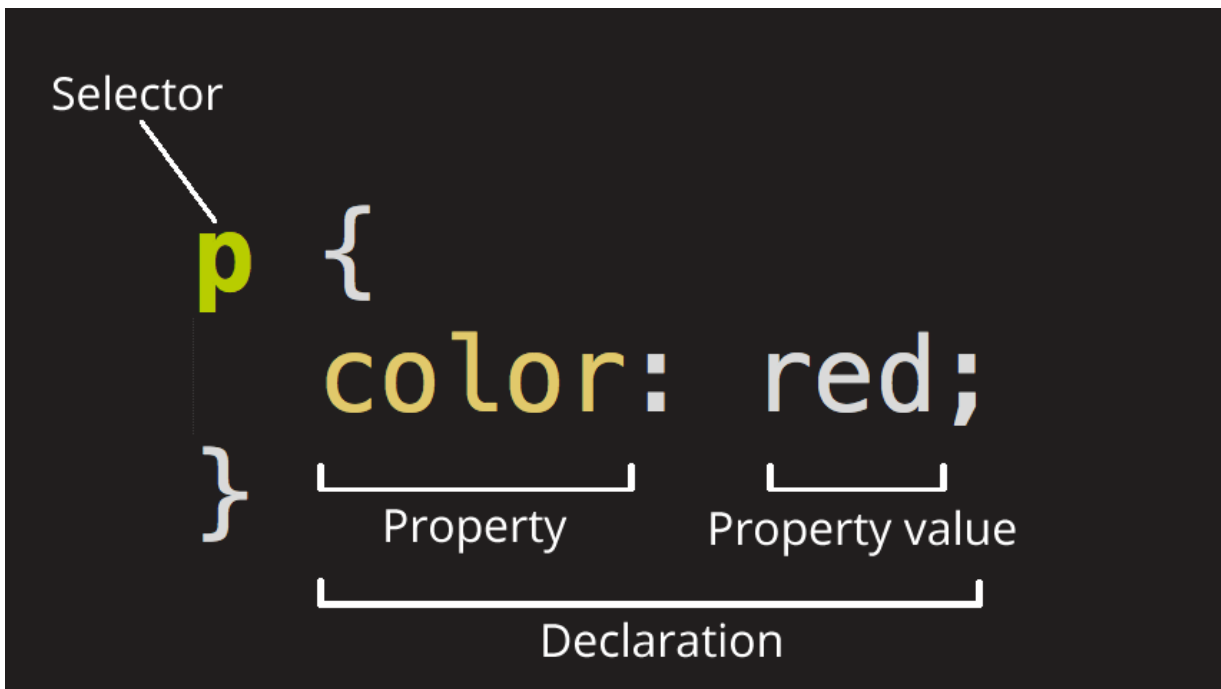
Materi Ajar

Pendahuluan

CSS merupakan kepanjangan dari *Cascading Style Sheets*. Dengan menggunakan CSS bersama HTML kita akan dapat memberi *style* pada elemen yang kita buat.

Sintaks Dasar

Pada dasarnya untuk menggunakan CSS kita memerlukan *selector* serta deklarasi.



- **Selector:** Penunjuk ke arah mana Elemen harus diberikan style.
- **Property:** Properti gaya yang akan digunakan.
- **Property Value:** Nilai yang akan diterapkan pada properti yang digunakan.
- **Deklarasi:** Terdiri dari **Property** dan **Property Value**, setiap deklarasi dipisahkan dengan titik koma (;).
- **Blok Deklarasi:** Terdiri dari beberapa deklarasi yang diletakkan kedalam kurung kurawal (`{ }`).
- Keseluruhan struktur tersebut disebut dengan ``rule`` atau aturan

Cara Menulis CSS

Inline Style

Inline Style merupakan cara penulisan CSS yang menuliskan deklarasi CSS langsung pada elemen yang ingin diberikan style. Dengan menggunakan inline style berarti kita tidak memerlukan menuliskan selector-nya, hal ini karena kita menuliskannya langsung ke elemen tersebut.

```
<p style="color: red;">
  Hai ini style dari CSS warna Merah 🙌
</p>
```

Eksperimen


1. Buatlah file html baru kemudian buatlah paragraf didalamnya.
2. Tambahkan atribut `style` pada elemen tersebut dengan nilai sebagai berikut:
`color: red; font-size: 32pt; font-family: 'Comic Sans MS'; background-color: green;`
3. Simpan file tersebut dan buka di browser.
4. Anda seharusnya dapat melihat paragraf berwarna merah berukuran besar dengan typeface `Comic Sans` dan background berwarna hijau.

Internal Style

Bayangkan kode HTML yang telah kita buat memiliki banyak elemen dan kemudian kita akan memberikan style dengan inline style, dimana setiap elemen kita menuliskan CSS-nya. Hal ini tentu akan membuat kita menulis ulang sesuatu yang seharusnya tidak diperlukan, untuk itulah adanya internal style. Dengan internal style kita dapat menuliskan CSS untuk banyak elemen sekaligus.

Untuk menuliskannya kita dapat menggunakan elemen `style` pada elemen `head`.

```
<html>
  <head>
    <title>Percobaan Inline Style</title>
    <style>
      p {
        color: red;
        background-color: yellow;
      }
    </style>
  </head>
  <body>
    ...
  </body>
</html>
```



```
<style>
  nav {
    position: fixed;
    top: 0;
    background-color: cyan;
  }
  p {
    color: red;
    background-color: yellow;
  }
</style>
```

Eksperimen

1. Buatlah file HTML baru, salin kode berikut:

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <main>
      <section>
        <h2>Inline Style</h2>
        <p>
          Dengan menggunakan inline style berarti kita tidak
          memerlukan menuliskan selector-nya, hal ini karena kita
          menuliskannya langsung ke elemen tersebut.
        </p>
      </section>

      <section>
        <h2>Internal Style</h2>
        <p>
          Dengan internal style kita dapat menuliskan CSS untuk
          banyak elemen sekaligus.
        </p>
      </section>

      <section>
        <h2>External Style</h2>
        <p>
          Kita dapat menuliskan kode CSS kedalam satu file
          tersendiri kemudian kita memanggilnya ke setiap file HTML kita.
        </p>
      </section>
    </main>
  </body>
</html>
```

2. Buka file HTML tersebut pada browser.
3. Tambahkan tag `style` didalam tag `head`.

4. Salin CSS berikut kedalam tag ``style`` yang telah dibuat sebelumnya

```
html {  
  font-family: Arial;  
}  
  
main {  
  width: 100%;  
  height: 100%;  
}  
  
section {  
  width: 60%;  
  margin: auto auto;  
  padding: 4px 12px;  
  border-radius: 8px;  
  box-shadow: 1px 3px 5px #02020280;  
}
```

5. Buka kembali file HTML tadi, kemudian coba bandingkan hasilnya dengan hasil sebelumnya (sebelum menggunakan tag ``style``).
6. Cobalah buat yang sama akan tetapi menggunakan *inline style*. Bandingkan antara penulisan inline style dengan internal style, mana yang lebih efisien?

External Style

Seiring berkembangnya proyek web kita maka akan semakin banyak pula halaman yang harus dibuat sehingga sudah tidak mungkin lagi apabila kita harus menyalin kode CSS kita ke setiap halaman. Dengan menggunakan External Style kita dapat menuliskan kode CSS kedalam satu file tersendiri kemudian kita memanggilnya ke setiap file HTML kita.

```
nav {  
  position: fixed;  
  top: 0;  
  background-color: cyan;  
}  
p {  
  color: red;  
  background-color: yellow;  
}
```

style.css

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

coba-external-style.html

Eksperimen

1. Ulangi langkah yang sama sampai no.2 pada eksperimen sebelumnya.
2. Buatlah file baru dengan nama `style.css`.
3. Salin kode CSS pada langkah no.4 pada eksperimen sebelumnya ke file `style.css` yang baru saja dibuat.

4. Salin elemen berikut kedalam tag `head`.

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

Selector Sederhana

Selector dalam CSS berfungsi sebagai penunjuk kemana style harus diterapkan.

Element Selector

Selector ini akan menerapkan ke elemen yang sesuai nama elemen yang sama.

```
p {  
  font-family: 'Comic Sans MD';  
}
```

ID Selector

Selector ini akan menerapkan style ke elemen dengan nilai id yang sama dengan nilai yang ada setelah simbol pagar (#).

```
#form-pemesanan {  
  margin: 12px;  
  border: 1px solid rgb(20, 20, 20);  
}
```

Pada contoh diatas style akan diterapkan ke elemen dengan atribut `id` bernilai `form-pemesanan`.

Class Selector

Selector ini akan menerapkan style ke elemen dengan nilai class yang sama dengan nilai yang ada setelah simbol titik (.).

```
.news-card {  
  padding: 12px 8px;  
  box-shadow: 1px 3px 5px rgba(20,20,20,25);  
}
```

Pada contoh diatas style akan diterapkan ke elemen dengan atribut `class` bernilai `news-card`.

Universal Selector

Selector ini akan menerapkan style ke keseluruhan elemen.

```
* {  
  color: red;  
}
```

Grouping Selector

Penulisan `rule` dalam CSS dapat diterapkan pada lebih dari 1 selector, untuk menuliskannya dapat dipisah dengan tanda koma (,).

```
h1, h2, h3, h4, h5, h6, .teks-judul {  
  margin: 0;  
  font-size: 1rem;  
}
```

Pada contoh diatas aturan tersebut akan diterapkan ke elemen h1 sampai h6 dan elemen dengan class `teks-judul`.

Selector lebih lanjut

- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Selectors
- https://www.w3schools.com/cssref/css_selectors.asp

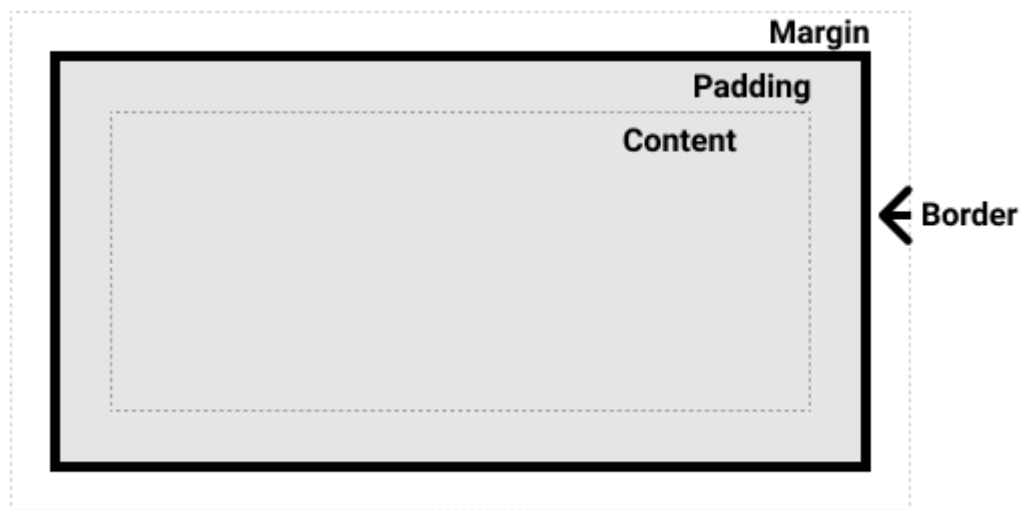
Tipografi

Berikut beberapa properti dalam CSS yang dapat digunakan untuk mengubah style tipografi:

- color
- font-size
- font-family
- font-weight
- text-align
- text-decoration
- text-shadow

Box Model

Box model merupakan konsep dasar tata letak elemen dalam CSS, box model merepresentasikan bagaimana setiap blok elemen bekerja. Blok pada HTML terdiri dari bagian berikut:



- Margin: Jarak dengan elemen lain.
- Padding: Jarak blok dengan elemen didalamnya.
- Border: Pembatas antara margin dan padding, border ini dapat kita letakkan ditengah pembatas tersebut atau kedalam ataupun keluar.
- Content: Konten ialah tempat elemen didalamnya berada.

Berikut beberapa properti box model pada CSS yang dapat digunakan:

- margin
- padding
- border
- border-radius
- box-sizing
- box-shadow
- background

Penempatan Posisi Elemen

Secara default setiap elemen pada web memiliki kedudukan sejajar dan akan berada pada urutan sesuai dengan peletakan penulisan kode HTML-nya. Dalam CSS sendiri terdapat 5 jenis bagaimana sebuah elemen harus ditempatkan.

Static

Static merupakan sifat default sebuah elemen. Menggunakan properti `top`, `right`, `bottom`, `left` tidak akan memengaruhi elemen ini.

```
.my-block {  
  position: static;  
}
```

Relative

`position: relative` akan diposisikan relatif terhadap letak normal (yang seharusnya). Menggunakan properti `top`, `right`, `bottom`, `left` akan diposisikan relatif terhadap letak normalnya.

```
.relative {  
  position: relative;  
  left: 30px;  
  top: 10px;  
  border: 3px solid #0F0F0F;  
}
```

Hasil

Ini bukan relative



Dapat dilihat dalam contoh diatas bahwa penggunaan posisi relatif yang dikombinasikan dengan top/right/bottom/left akan bergeser dari tempat normal (yang seharusnya).

Absolute

Penggunaan `position: absolute;` akan membuat elemen dihapus dari posisi normal (yang seharusnya) dan akan ditempatkan relatif terhadap elemen diatasnya yang memiliki posisi `relative`. Jika tidak ada maka akan ditempatkan relatif terhadap halaman layar. Kemudian posisinya akan dijumlahkan dengan properti top/right/bottom/left.

```
<html>
  <head>
    <style>
      .relatif {
        position: relative;
        height: 160px;
      }
      .absolut {
        position: absolute;
        bottom: 10px;
        right: 24px;
      }
      .relatif, .absolut {
        border: 3px solid red;
      }
    </style>
  </head>
  <body>
    <main>
      <div class="relatif">
        <span>Ini relatif</span>

        <div class="absolut">
          Ini absolute
        </div>
      </div>
    </main>
  </body>
</html>
```

Hasil

Ini relatif

Ini absolute

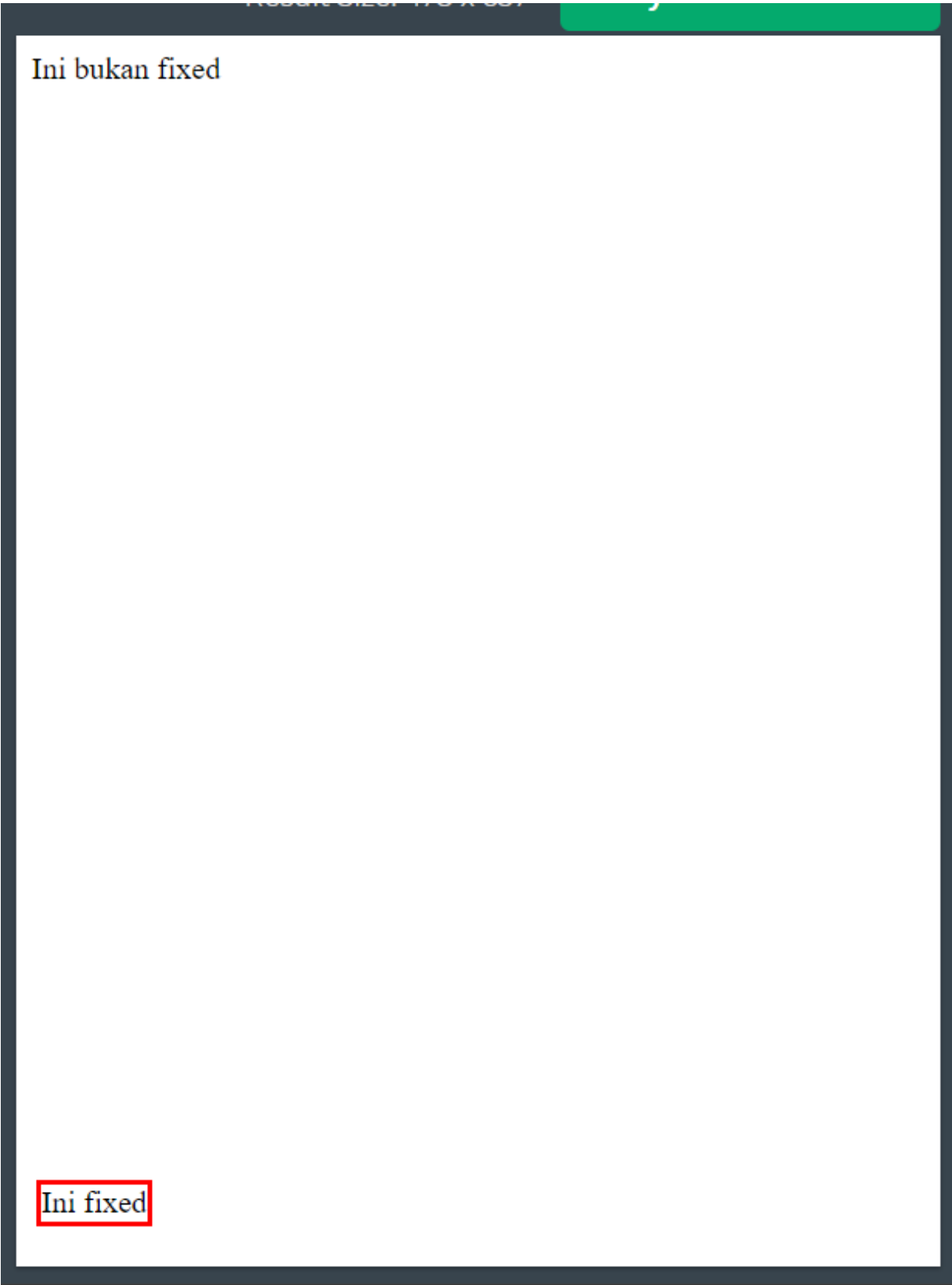
Fixed

Sama halnya dengan `absolute`, hanya saja `fixed` akan relatif terhadap halaman layar, sehingga apabila halaman di-scroll maka akan tetap terlihat di halaman layar, tidak tenggelam.

```
<html>
  <head>
    <style>
      .fixed {
        position: fixed;
        bottom: 20px;
        left: 10px;
        border: 3px solid red;
      }
    </style>
  </head>
  <body>
    <div>
      Ini bukan fixed
    </div>

    <div class="fixed">
      Ini fixed
    </div>
  </body>
</html>
```

Hasil



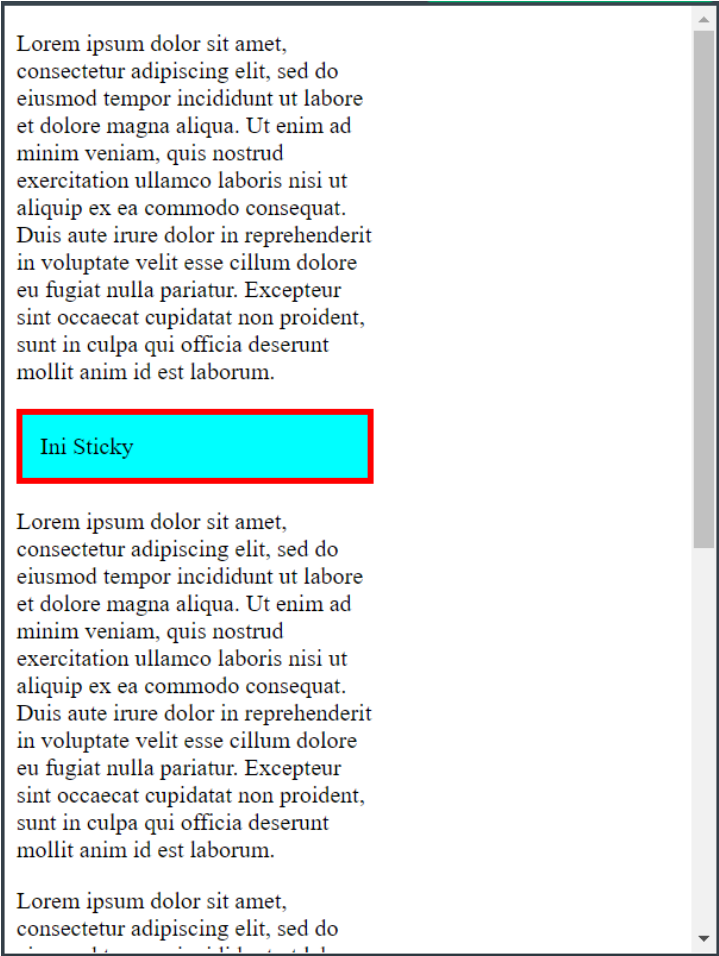
Sticky
`position: sticky;` merupakan kombinasi antara `relative` dan `fixed`, posisi fixed ini akan aktif apabila user telah menye-scroll sampai ke elemen tersebut. Gunakan properti top/right/bottom/left untuk memanfaatkan posisi fixed-nya.

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      main {
        max-width: 15rem;
      }
      .sticky {
        position: sticky;
        top: 0;
        left: 0;
        right: 0;
        padding: 12px;
        background-color: cyan;
        border: 4px solid red;
      }
    </style>
  </head>
  <body>
    <main>
      <p>...</p>

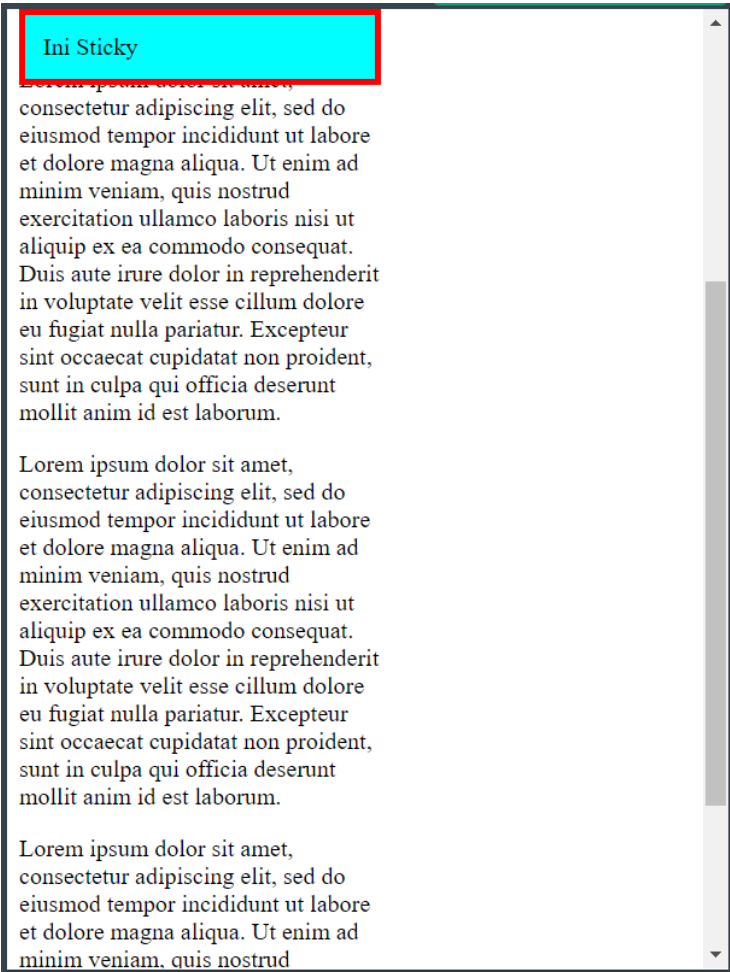
      <div class="sticky">
        Ini Sticky
      </div>

      <p>...</p>
      <p>...</p>
      <p>...</p>
    </main>
  </body>
</html>
```

Hasil sebelum user scroll sampai ke elemen



Hasil setelah user scroll mencapai elemen sticky



Properti CSS Lainnya

- <https://developer.mozilla.org/en-US/docs/Web/CSS/Reference#index>
- <https://www.w3schools.com/cssref/default.asp>

Selector Lanjutan

Pseudo-classes Selector

Merupakan selector spesial untuk keadaan tertentu pada suatu elemen.

:hover

Rule CSS akan aktif ketika cursor sedang berada diatas elemen.

```
a:hover {  
  background-color: yellow;  
}
```

Hasil

[Ini Link](#)

Ketika mouse berada diatas elemen

[Ini Link](#)

:active

Rule akan aktif ketika user meng-klik elemen.

```
a:active {  
  color: white;  
  background-color: blue;  
}
```

Hasil, ketika user klik elemen

[Ini Link](#)

:focus

Rule akan aktif ketika user sedang mencoba berinteraksi dengan elemen. Semisal elemen input berarti ketika user sedang mencoba memasukkan teks.

```
input:focus {  
  outline: none;  
  border: 2px solid blue;  
}
```

Hasil

Dan banyak lagi...

- <https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-classes>

Pseudo-element Selector

Pseudo-element merupakan selector untuk memilih sebagian dari bagian suatu elemen, semisal dari suatu paragraf kita dapat mengambil baris pertama saja dari paragraf dengan selector `::first-line`.

::first-child, ::last-child

`::first-child` digunakan untuk memilih elemen pertama pada elemen yang sama dan satu tingkat, sebaliknya pada `::last-child`.

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      p:first-child {
        color: red;
      }
    </style>
  </head>
  <body>
    <main>
      <p>...</p>
      <p>...</p>
      <p>...</p>
    </main>
  </body>
</html>
```

Hasil

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

::before, ::after

Selector ini digunakan untuk menambahkan elemen, `::before` akan menambahkan elemen sebelum elemen sebaliknya `::after` akan menambahkan elemen setelahnya.

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      .nyobain::before,
      .nyobain::after {
        background-color: cyan;
      }

      .nyobain::before {
        content: 'sebelum';
      }

      .nyobain::after {
        content: 'setelah';
      }
    </style>
  </head>
  <body>
    <main>
      <div class="nyobain">Ini Elemen asli</div>
    </main>
  </body>
</html>
```

Hasil

sebelumIni Elemen aslisetelah

Dan banyak lagi...

- <https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-elements>