WEB PROGRAMMING

MODUL PRAKTIKUM 2 CSS

DAFTAR ISI

[1 Pengenalan CSS 3](#_Toc49649813)

[1.1 Tentang CSS (*Cascading Style Sheet*) 3](#_Toc49649814)

[1.2 Kegunaan CSS 3](#_Toc49649815)

[1.3 Aturan Penulisan CSS 3](#_Toc49649816)

[1.4 Cara Penulisan CSS 4](#_Toc49649817)

[1.4.1 Atribut rel 6](#_Toc49649818)

[1.4.2 Atribut href 6](#_Toc49649819)

[1.4.3 Atribut media 6](#_Toc49649820)

[1.5 Komentar pada CSS 7](#_Toc49649821)

[1.6 Macam - macam Selector 7](#_Toc49649822)

[*1.6.1* *Universal Selector* 7](#_Toc49649823)

[1.6.2 Tag / Element HTML 7](#_Toc49649824)

[*1.6.3* *Class Selector* 8](#_Toc49649825)

[*1.6.4* *ID Selector* 10](#_Toc49649826)

[*1.6.5* *Child Selector* 11](#_Toc49649827)

[*1.6.6* *Descendant selector* 12](#_Toc49649828)

[*1.6.7* *Adjacent Sibling Selector* 13](#_Toc49649829)

[*1.6.8* *General Sibling Selector* 14](#_Toc49649830)

[2 Font 16](#_Toc49649831)

[2.1 Properti font-family 18](#_Toc49649832)

[*2.2* Properti *font-size* 19](#_Toc49649833)

[*2.3* Properti *font-weight* 21](#_Toc49649834)

[2.4 Properti *font-style* 22](#_Toc49649835)

[2.5 Properti font-variant 23](#_Toc49649836)

[3 Teks 25](#_Toc49649837)

[3.1 Properti color 25](#_Toc49649838)

[3.2 Properti text-align 27](#_Toc49649839)

[3.3 Properti vertical-align 28](#_Toc49649840)

[3.4 Properti text-decoration 30](#_Toc49649841)

[3.5 Properti text-indent 32](#_Toc49649842)

[4 Box Model 33](#_Toc49649843)

[4.1 Ilustrasi box model 34](#_Toc49649844)

[5 Border 36](#_Toc49649845)

[5.1 Properti Border 36](#_Toc49649846)

[5.2 Properti border-width 36](#_Toc49649847)

[*5.3* Properti *border-style* 36](#_Toc49649848)

[*5.4* Properti *border-color* 37](#_Toc49649849)

[*5.5* Properti *border (singkat)* 38](#_Toc49649850)

[*6* *Margin* dan *Padding* 42](#_Toc49649851)

[*6.1* Properti *Padding* 42](#_Toc49649852)

[*6.2* Properti *margin* 44](#_Toc49649853)

[6.3 Properti text-shadow 46](#_Toc49649854)

[6.4 Properti text-transform 47](#_Toc49649855)

[6.5 Properti letter-spacing 49](#_Toc49649856)

[6.6 Properti word-spacing 50](#_Toc49649857)

[6.7 Properti white-space 51](#_Toc49649858)

[6.8 Properti direction 55](#_Toc49649859)

[7 CSS Layout 57](#_Toc49649860)

[7.1 Block Level Elements 57](#_Toc49649861)

[7.2 Inline Elements 57](#_Toc49649862)

[7.3 Properti Display : none; 57](#_Toc49649863)

[7.4 Override The Default Display Value 58](#_Toc49649864)

[7.5 Properti visibility : hidden; 60](#_Toc49649865)

[8 CSS Layout Float & Clear 62](#_Toc49649866)

[8.1 Properti FLoat 62](#_Toc49649867)

[8.2 Properti Clear 64](#_Toc49649868)

# Pengenalan CSS

## Tentang CSS (*Cascading Style Sheet*)

CSS (*Cascading Style Sheet*) secara sederhana adalah sebuah metode yang digunakan untuk mempersingkat penulisan tag HTML, seperti *font*, *color*, *text* dan tabel menjadi lebih ringkas sehingga tidak terjadi pengulangan penulisan.

CSS digunakan untuk mengatur tampilan dokumen. Dengan adanya CSS, memungkinkan Kita untuk menampilkan halaman yag sama dengan format berbeda.

CSS merupakan teknologi internet yang direkomendasikan oleh *World Wide Web Consortium* atau W3C pada tahun 1996.

Saat ini terdapat 3 versi CSS (CSS1, CSS2, CSS3) mayoritas property - property yang digunakan adalah CSS2 karena property CSS3 masih dapat berubah - ubah selama pengembangan.

## Kegunaan CSS

Beberapa kegunaan CSS diantaranya :

1. Mempersingkat penulisan tag HTML

Penulisan tag dengan property dan nilai yang sama tidak perlu dituliskan pada setiap tag HTML.

1. Mempercepat proses rendering atau pembacaan HTML karena tidak terdapat pengulangan penulisan.
2. Mudah untuk memelihara (*maintain*) skrip : karena CSS dapat dibuat terpisah maka tidak perlu merombak semua elemen atau property dalam HTML.
3. CSS dapat melakukan apa yang tidak bisa dilakukan oleh HTML seperti memberikan warna pada *input box* atau *scrollbar*.

## Aturan Penulisan CSS

CSS memiliki aturan dalam penulisannya, yang terdiri atas ***Selector*** dan ***Declaration.***

* **Selector :** mengidikasikan elemen atau elemen - elemen yang akan di deklarasikan.
* **Declaration :** untuk mendeskripsikan property dan nilai yang akan digunakan pada *selector.*

****

Gambar 1 Selector dan Declaration pada CSS.

CSS *Declaration* yang berada didalam kurung kurawal terdiri dari 2 (dua) bagian ***property*** dan ***Value*** yang dipisahkan oleh tanda titik dua “ : ”. Kita dapat menentukan beberapa property dalam satu deklarasi, masing - masing dipisahkan dengan tanda titik koma “ ; “.

A picture containing text

Description automatically generated

Gambar 2 Property dan value pada CSS.

## Cara Penulisan CSS

Cara penulisan skrip CSS dibagi menjadi 3 (tiga) bagian, yaitu :

1. ***Inline Style Sheet***

Penulisan skrip CSS didalam elemen HTML. Untuk metode ini fungsinya hamper sama dengan menuliskan property pada tag HTML, penggunaannya untuk hal - hal tertentu saja. Misalkan memberi warna hijau pada tag “<p>” maka cara penulisannya sebagai berikut :

<p style=”color:green”>

Penulisan skrip CSS dilakukan pada tag pembuka suatu element html. Property ***style*** akan men-support semua property yang di-support oleh CSS.

1. ***Embedded Style Sheet***

Penulisan skrip CSS didalam dokumen HTML. Metode ini menggunakan tag **<style>** dan **</style>**

Contoh :

<style type=”text/css>

p {color : red}

</style>

Biasanya penulisan skrip CSS ini dilakukan pada bagian elemen “head”.

1. ***Linked Style Sheet / External Style Sheet***

Penulisan skrip CSS di halaman yang berbeda atau terpisah dari HTML. Jadi, Kita tinggal melakukan link ke file CSS yang telah dibuat. Untuk metode ini, menggunakan tag <link rel> yang ditempatkan pada bagian tag <head>. Metode ini paling banyak digunakan dalam penulisan CSS. *Linked style sheet* merupakan metode penulisan yang paling banyak dipakai pada pengembangan website.

Beberapa keuntungan menggunakan aturan *style* ini diantaranya :

* 1. Menghemat Kita mengulangi aturan *style* yang sama di setiap halaman.
  2. Dapat mengubah tampilan beberapa halaman dengan mengubah hanya pada *style sheet -* nya saja daripada masing-masing halaman. Ini berarti lebih mudah untuk memperbarui website Anda jika Anda ingin, misalnya, mengubah gaya font yang digunakan dalam semua judul atau mengubah warna semua tautan.
  3. Setelah pengunjung ke situs Anda mengunduh CSS dengan halaman pertama situs Anda yang menggunakannya, halaman berikutnya akan lebih cepat untuk memuat (karena browser menyimpan salinan style sheet dan aturan tidak perlu diunduh untuk setiap halaman). Ini juga mengurangi ketegangan pada server (komputer yang mengirim halaman web ke orang yang melihat situs) karena halaman yang dikirim lebih kecil.
  4. *Style sheet* dapat bertindak sebagai *template style* untuk membantu penulis yang berbeda mencapai *style sheet* dokumen yang sama tanpa mempelajari semua pengaturan *style* individu.
  5. Karena halaman web tidak mengandung aturan gaya (*style rules*), berbagai *style sheet* dapat dilampirkan ke dokumen yang sama. Jadi, Anda dapat menggunakan dokumen HTML yang sama dengan satu *style sheet* saat *viewer* berada di komputer desktop, *style sheet* lain saat pengguna memiliki perangkat genggam (*mobile*), *style sheet* lain saat halaman sedang dicetak (*print*), *style sheet* lain saat halaman sedang dilihat di TV, dan sebagainya. Anda dapat menggunakan kembali dokumen yang sama dengan *style sheet* berbeda untuk kebutuhan pengunjung yang berbeda.
  6. *Style sheet* dapat mengimpor dan menggunakan *style* dari *style sheet* lain, memungkinkan untuk pengembangan modular dan penggunaan kembali yang baik.
  7. Jika *style sheet* dihapus, Anda dapat membuat situs lebih mudah diakses bagi mereka yang memiliki keterbatasan penglihatan karena Anda tidak lagi mengendalikan font dan skema warna.

Oleh karena itu, cukup adil untuk mengatakan bahwa setiap kali Kita menulis seluruh situs, Anda harus menggunakan style sheet eksternal untuk mengontrol penyajiannya (daripada meletakkan aturan CSS di halaman web individual).

### Atribut rel

Atribut rel diperlukan dan menentukan hubungan antara dokumen (*relationship*) yang berisi tautan dan dokumen yang ditautkan. Nilai yang digunakan untuk bekerja dengan style sheet adalah ***stylesheet***:

rel=”stylesheet”

### Atribut href

Atribut ***href*** menentukan URL untuk dokumen yang ditautkan :

href = "../ stylesheets / interface.css"

Nilai dari atribut ini dapat berupa URL absolut atau relatif (dibahas pada buku panduan definitif html5), tetapi biasanya merupakan URL relatif karena style sheet adalah bagian dari situs.

### Atribut media

Atribut media menentukan perangkat output yang dimaksudkan untuk digunakan dengan dokumen:

media = "screen"

Meskipun atribut ini tidak selalu digunakan, ini penting karena orang mengakses Internet dengan cara yang berbeda menggunakan perangkat yang berbeda. Tabel 7-1 menunjukkan nilai yang mungkin.

Tabel 1 Atribut media.

|  |  |
| --- | --- |
| Nilai (value) | Penggunaan |
| screen | Layar komputer (seperti komputer desktop dan laptop) |
| tty | Media dengan kisi karakter pitch tetap, seperti teletype, terminal, atau perangkat portabel dengan kemampuan tampilan terbatas |
| Tv | Perangkat TV dengan resolusi rendah, layar warna, dan kemampuan terbatas untuk menggulir ke bawah halaman |
| print | Dokumen tercetak, yang kadang-kadang disebut media paged (dan dokumen ditampilkan di layar dalam mode pratinjau cetak (*preview mode*) |
| projection | Projector |
| handheld | Perangkat genggam, yang sering memiliki layar kecil, bergantung pada grafik yang dipetakan, dan memiliki bandwidth terbatas |
| braille | Perangkat umpan balik sentuhan Braille |
| embossed | Printer beranda braille |
| speech | Penyintesis ucapan |
| all | Cocok untuk semua perangkat |

## Komentar pada CSS

Sama seperti Bahasa pemograman lainnya komentar akan diberikan paa skrip untuk menjelaskan sesuatu. Komentar pada CSS hampir sama dengan komentar pada Bahasa C atau C++, yaitu menggunakan :

/\* isi komentar \*/

Contoh :

p {

color : red; /\* memberikan warna merah pada paragraf\*/

}

## Macam - macam Selector

Sebelum melanjutkan untuk melihat lebih banyak properti, Anda perlu melihat beberapa masalah yang lebih mendasar. Mulailah dengan melihat bagaimana Anda dapat menggunakan jenis *selector* yang berbeda untuk menentukan elemen mana yang menjadi aturan aturan *style sheet*. Sebenarnya ada beberapa cara untuk melakukan ini, tidak hanya dengan menggunakan nama elemen seperti yang telah Kita lihat sejauh ini dalam bab ini (yang, kebetulan, dikenal sebagai *selector* sederhana) atau menggunakan nilai atribut kelas atau atribut id. Pelajari caranya di beberapa bagian berikutnya.

### *Universal Selector*

*Selector* universal adalah tanda bintang; itu seperti *wildcard* dan cocok dengan semua jenis elemen dalam dokumen.

\* { }

Jika Anda ingin aturan diterapkan ke semua elemen, Anda dapat menggunakan *selector* ini. Kadang-kadang digunakan untuk nilai-nilai *default* yang berlaku untuk seluruh dokumen (seperti ***font-family*** dan ukuran ***font***) kecuali *selector* lain yang lebih spesifik menunjukkan bahwa elemen harus menggunakan nilai yang berbeda untuk properti yang sama.

Ini sedikit berbeda dari menerapkan gaya *default* ke elemen ***<body>*** karena pemilih universal berlaku untuk setiap elemen dan tidak bergantung pada properti yang diwarisi dari aturan yang berlaku untuk elemen ***<body>.***

### Tag / Element HTML

Penggunaan tag yang terdapat pada HTML. Setiap tag yang ada dalam HTML dapat dijadikan sebagai *selector.*

Contoh :

**h1 {color : red; }**

### *Class Selector*

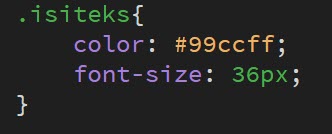
*Class selector* memungkinkan Anda untuk mencocokkan aturan dengan yang membawa atrribute ***class*** yang nilainya cocok dengan yang Anda tentukan di c*lass selector*.

Sebagai contoh, pada tag “<p>” akan Kita tambahkan atribut class dengan nama *isiteks* seperti berikut :

<p class=”isiteks”>ini adalah isiteks class paragraph</p>

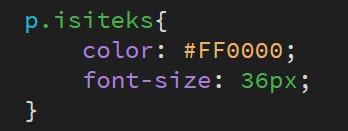
Penggunaan c*lass selector* akan diawali dengan tanda titik “.” Diawal penulisannya. Kita dapat menggunakan c*lass selector* dengan 2 (dua) cara.

1. Kita dapat memberikan aturan penulisan CSS pada semua element yang memiliki atribut class “isiteks”.



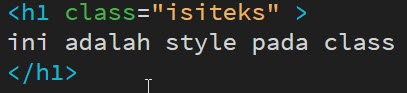
Gambar 3 Class CSS.

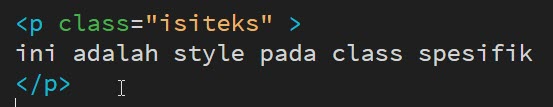
1. Kita dapat memberikan aturan penulisan CSS hanya pada spesifik elemen yang sudah diberikan class tertentu saja.



Gambar 4 Class Spesifik CSS.

Kemudian untuk mengimplementasikan style pada class yang sudah dibuat, pada dokumen html diantara tag **body** dapatdibuat sebagai berikut :





Gambar 5 isi script implementasi *class style* pada dokument HTML.

Maka pada hasil di browser dapat terlihat sebagai berikut :



Gambar 6 Tampilan Browser Implementasi Class CSS.

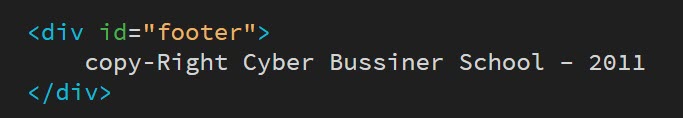
### *ID Selector*

Hampir sama dengan class selector, *ID selector* bersifat unik dan dapat diterapkan untuk hampir semua tag HTML, tetapi penggunaanya hanya sekali dalam satu halaman untuk satu tag HTML tertentu. Penggunaan ID selector menggunakan awalan tanda pagar “#” untuk inisialisasi sebuah nama id yang sudah dibuat pada dokumen HTML.



Gambar 7 Style ID.

pada CSS ditulis :



Gambar 8 Implementasi ID Style pada HTML.

Sekali lagi di tekankan selector ID digunakan hanya untuk 1 elemen pada satu

halaman web. Misalnya saja ID **#footer** diatas hanya digunakan sekali karena

dalam satu halaman web hanya ada 1 id ***footer***.

Hasil pada browser :

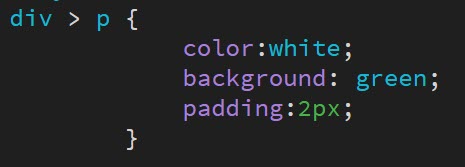


Gambar 9 Hasil Implementasi Style Id.

### *Child Selector*

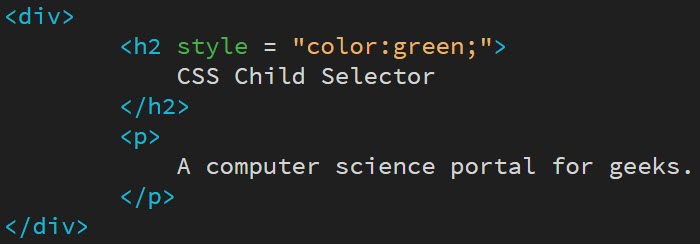
*Child selector* digunakan untuk mencocokkan semua elemen yang merupakan anak dari elemen yang ditentukan. Ini memberi hubungan antara dua elemen. Selector **elemen > elemen** memilih elemen yang merupakan anak-anak dari orang tua tertentu. Operan di sisi kiri “>” adalah induk dan operan di sebelah kanan adalah elemen anak-anak.

Style CSS :



Gambar 10 Style CSS Child Selector.

Dokumen Html :



Gambar 11 HTML Style Child Selector.

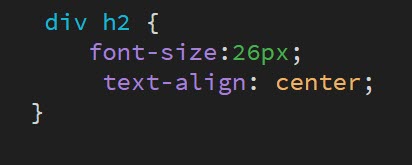
****

Gambar 12 Hasil css child selector.

### *Descendant selector*

*Selector Descendant* digunakan untuk memilih semua elemen yang merupakan anak dari elemen (bukan elemen tertentu). Ini memilih elemen di dalam elemen yaitu menggabungkan dua penyeleksi sehingga elemen yang cocok dengan *selector* kedua dipilih jika mereka memiliki elemen leluhur yang cocok dengan *selector* pertama.

Style CSS :



Gambar 13 Style Descendant Selector CSS.

Dokumen HTML :



Gambar 14 Dokumen HTML Descendant Selector.



Gambar 15 hasil penggunaan descendand selector.

### *Adjacent Sibling Selector*

Sebuah *adjacent sibling selector* digunanakan untuk memilih diantara 2 pasang selector dan dipilih yang kedua atau selanjutnya.

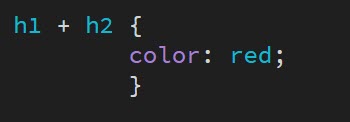
Jika, x, y dan z adalah tiga elemen HTML dan y dan z berada bersebelahan di dalam x, maka y dan z disebut sebagai *adjacent sibling selector.*

Saat menulis *adjacent sibling selector*, *selector* harus dipisahkan dengan kombinator "+".

Misalnya jika Anda ingin membuat paragraf pertama setelah setiap level 1 dengan gaya yang berbeda dari elemen <p> lainnya, Anda dapat menggunakan *adjacent sibling selector* seperti itu untuk menentukan aturan untuk hanya elemen <p> pertama yang muncul setelah setiap elemen <h1>.

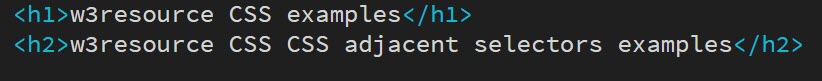
Contoh penggunaan *adjacent sibling selector :*

CSS Style :



Gambar 16 CSS Adjacent sibling selector.

Dokumen HTML :



Gambar 17 HTML Adjacent Sibling Selector.

A close up of a logo

Description automatically generated

Gambar 18 contoh hasil adjacent sibling selector.

Jika Kita menemukan h2 kembali tanpa h1 pada paragraf lainnya maka mereka bukan adjacent sibling selector karena selector sebelumnya bukan h1.

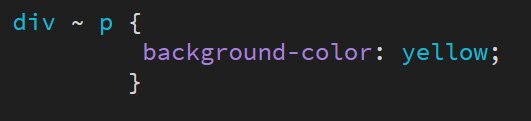
### *General Sibling Selector*

*General sibling selector* memilih semua elemen yang merupakan saudara kandung dari elemen yang ditentukan.

Contoh berikut memilih semua elemen <p> yang merupakan saudara dari elemen <div>:

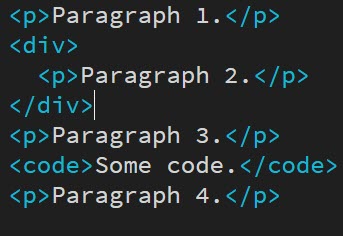
Saat menulis general sibling selector selector harus dipisahkan dengan kombinator “~”

Style CSS :

****

Gambar 19 CSS Style General Sibling Selector.

Dokument HTML :



Gambar 20 HTML Style General Sibling Selector.

**A screenshot of a cell phone

Description automatically generated**

Gambar 21 Hasil pada Browser general sibling selector.

# Font

Selanjutnya Kita akan mempelajari mengenai kontrol teks saya akan membagi menjadi 2 (dua) bab yang membahas mengenai kontrol teks ini. bab ini adalah bagian yang pertama yaitu kontrol teks yang langsung mempengaruhi font dan penampilannya. Misalnya huruf tebal, cetak miring dan ukuran teks.

Berikut adalah property font yang dapat mempengaruhi secara langsung (*directly*) pada sebuah font teks.

Tabel 3 Properti Font.

|  |  |
| --- | --- |
| Properti | Kegunaan |
| font | Memungkinkan Anda untuk menggabungkan beberapa properti berikut ini menjadi satu. |
| font-family | Menentukan jenis huruf atau keluarga font yang harus digunakan. |
| font-size | Menentukan ukuran font. |
| font-weight | Menentukan apakah font harus normal atau tebal. |
| font-style | Menentukan apakah font harus normal, miring, atau miring. |
| font-variant | Menentukan apakah font harus berukuran normal atau kecil. |

Sebelum melihat properti ini secara detail, ada baiknya Anda memahami beberapa istilah kunci yang digunakan dalam tipografi.

Mungkin yang paling penting, **font** tidak sama dengan jenis huruf (***typeface***):

* jenis huruf (*typeface*) adalah keluarga font, seperti keluarga Arial.
* Font adalah anggota spesifik keluarga itu, seperti huruf tebal 12 poin Arial.

Anda sering melihat istilah yang digunakan secara bergantian, tetapi perlu diperhatikan perbedaannya.

*Typeface* cenderung menjadi salah satu dari dua grup: font **serif** dan **sans-serif**. Font serif memiliki ikal tambahan pada huruf. Sebagai contoh, pada gambar dibawah ini, gambar pertama berisi serif di bagian atas huruf dan di bagian bawah huruf, sedangkan font sans-serif memiliki ujung lurus ke huruf, seperti pada contoh kedua.

Gaya umum ketiga dari jenis huruf adalah font serif **monospace**. Setiap huruf dalam font monospasi memiliki lebar yang sama, sedangkan font nonmonospasi memiliki lebar berbeda untuk huruf yang berbeda. (Misalnya, dalam font serif dan sans-serif, huruf l cenderung lebih sempit daripada huruf m.)



Gambar 23 Font Serif, Sans-serif, Monospace.\

Font serif umumnya dianggap lebih mudah dibaca untuk sejumlah besar teks yang dicetak. Tetapi di Internet banyak orang menemukan font serif lebih sulit dibaca untuk waktu yang lama, terutama karena resolusi layar komputer tidak sebagus dokumen cetak, yang membuat font sans-serif yang kurang rinci lebih mudah dibaca.

Agar Anda dapat mempelajari properti yang memengaruhi font, sebagian besar contoh di bagian berikut menggunakan struktur yang serupa. Paragraf teks akan diulangi, dan setiap elemen <p> membawa atribut kelas dengan nilai yang berbeda, misalnya:

A screenshot of a cell phone

Description automatically generated

Gambar 24 Script HTML & CSS Font Family.

Anda kemudian dapat melihat bagaimana properti berbeda memengaruhi setiap elemen <p> dengan menulis aturan terpisah untuk setiap paragraf. Anda dapat menggunakan nilai atribut kelas di pemilih CSS untuk membuat aturan yang berlaku hanya untuk satu elemen <p> pada suatu waktu.

A screenshot of a cell phone

Description automatically generated

Gambar 25 Hasil pada browser gambar 23.

## Properti font-family

Properti font-family memungkinkan Anda untuk menentukan jenis huruf yang harus digunakan untuk teks apa pun di dalam elemen yang diterapkan aturan CSS.

Saat memilih tipografi, Anda harus tahu bahwa tanpa menggunakan arahan @ font-face, browser hanya dapat menampilkan teks HTML dalam font yang Anda tentukan jika jenis huruf itu diinstal pada komputer itu. Jadi, jika Anda menentukan font seperti ***Futura*** atau ***Garamond***, dan saya tidak memilikinya di komputer saya, saya akan melihat teks dalam font yang berbeda bukan yang Anda tentukan.

Inilah sebabnya, jika Anda melihat pilihan situs web, sebagian besar sangat bergantung pada pilihan kecil tipografi yang dipasang pada sebagian besar komputer yang mengakses web, khususnya Arial, Kurir / Kurir Baru, Georgia, Times / Times New Roman, dan Verdana . (Dari daftar ini, Arial dan Verdana sangat populer karena dianggap mudah dibaca secara online.)

Untuk membantu masalah, Anda dapat menentukan daftar tipografi sehingga jika pengguna tidak memiliki pilihan jenis huruf yang diinstal pada komputer mereka, browser dapat mencoba menampilkan teks dalam pilihan kedua atau ketiga. Setiap jenis huruf dalam daftar dipisahkan oleh koma, dan jika nama berisi **spasi** (seperti ***times new roman*** atau ***courier new***), Anda harus menempatkan nama jenis huruf dalam tanda kutip ganda (gambar 23

Anda mungkin memperhatikan bahwa setiap daftar tipografi pada contoh sebelumnya diakhiri dengan apa yang disebut nama **font generik** (sans-serif, serif, dan monospace). Gagasan di balik ini adalah bahwa setiap komputer akan memiliki font yang sesuai dengan salah satu dari lima grup font generik (**sans-serif, serif, monospace, cursive, dan fantasy**), dan jika tidak dapat menemukan tipografi yang telah Anda tentukan, dapat menggunakan pilihan font yang sesuai dengan grup font umum.

Berikut adalah beberapa nama font yang umum.

Tabel 4 Nama Font yang Umum.

|  |  |  |
| --- | --- | --- |
| Nama Font Umum | Tipe font | Contoh |
| Serif | Font dengan serif | Times |
| Sans-serif | Font tanpa serif | Arial |
| Monospace | Font lebar fix | Courier |
| Cursive | Font yang meniru tulisan tangan | Comis sans |
| fantasy | Font dekoratif untuk judul dan sebagainya | Impact |

Satu hal yang perlu dipertimbangkan ketika memilih daftar font adalah bahwa setiap font dapat memiliki ketinggian atau lebar yang berbeda, jadi Anda mungkin ingin memilih daftar font yang memiliki ukuran yang sama. (Kalau tidak, tata letak bisa terlihat berbeda dengan yang Anda harapkan.) Misalnya, Courier New cukup pendek dan lebar, jadi jika ini adalah pilihan pertama Anda, itu tidak baik untuk memiliki Dampak sebagai pilihan kedua karena Dampak cukup tinggi dan sempit.

Jika Anda ingin menggunakan jenis huruf tertentu, Anda harus banyak mencoba - coba dan mempelajari mengenai web font lebih mendalam lagi.

## Properti *font-size*

**Properti font-size** digunakan untuk menentukan ukuran font. Biasanya nilai untuk property ini ditentukan dalam ukuran pixel (px).

Namun Kita dapat memberikan nilai dengan banyak cara :

* ***Length*** : selain pixel ada beberapa satuan panjang lain yag dapat digunakan pada ukuran font.

**px, em, ex, pt, in, cm, pc, mm, rem, vw, vh**

* ***Ukuran absolut*** : masing-masing nilai ini sesuai dengan ukuran tetap :

**xx-small, x-small, small, medium, large, x-large, xx-large**

* ***Ukuran relative*** : nilai ini relatif terhadap teks diseKitarnya :

**smaller, larger**

* ***Persentase*** : dihitung sebagai proporsi dari elemen induk :

**2%, 10%, 25% 100%**

Berikut adalah contoh beberapa cara untuk menentukan ukuran font pada css :

Pada contoh ini saya membuat sebuah document html dengan 4 paragraf yang memiliki class yang berbeda. Pada class tersebut ditentukan style css sebagai berikut :

A screenshot of a video game

Description automatically generated

Gambar 26 CSS style font-size.

Yang menghasilkan format dokumen html sebagai berikut :

A screenshot of a social media post

Description automatically generated

Gambar 27 html dokumen font-size.

## Properti *font-weight*

Property ini digunakan untuk membuat variasi font yang berbeda seperti cetak tebal dan cetak miring. Pada sebuah browser cenderung menggunakan sebuah algoritma untuk membuat versi font terlihat lebih tebal atau lebih tipis. Inilah tujuan dari property font-weight.

Nilai yang mungkin untuk font-weight adalah :

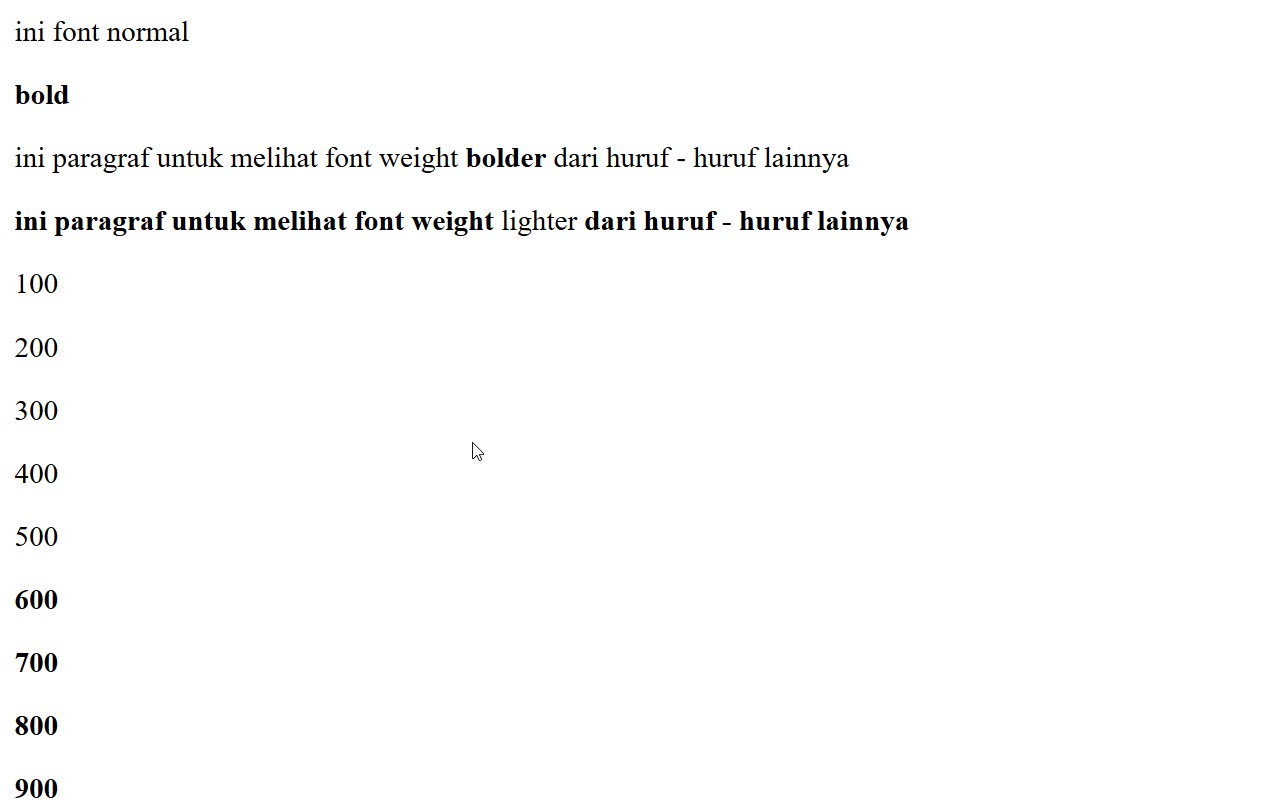
**normal bold bolder lighter 100 200 300 400 500 600 700 800 900**

berikut adalah contoh penggunaan property font-weight pada css :

|  |
| --- |
| p.normal{ font-weight: normal; }  p.bold{ font-weight: bold; }  p.bold span{font-weight: lighter;}  p.normal span{font-weight: bolder;}  p.satu{font-weight: 100;}  p.dua{font-weight: 200;}  p.tiga{font-weight: 300;}  p.empat{font-weight: 400;}  p.lima{font-weight: 500;}  p.enam{font-weight: 600;}  p.tujuh{font-weight: 700;}  p.delapan{font-weight: 800;}  p.sembilan{font-weight: 900;} |

Gambar 28 Script CSS font-weight.

Hasil yang dapat dilihat pada browser sesuai dengan dokumen html yang sudah dibuat sebelumnya :



Gambar 29 Hasil font-weight css.

## Properti *font-style*

Property ini digunakan untuk mengatur bentuk style pada font. Apakah secara normal, italic, atau oblique. Namun style bold tidak dikategorikan ke property font-style karena style bold dikategorikan ke property weight, hal ini berbeda dengan HTML.

Dalam tipografi, versi italic dari sebuah font biasanya akan menjadi versi font khusus berdasarkan kaligrafi, sedangkan versi oblique dari font akan mengambil versi normal font yang sudutnya dimiringkan.

Berikut adalah contoh penggunaan property font-style :

A close up of text on a black background

Description automatically generated

Gambar 30 css font-style.

Hasil yang terlihat pada browser :

A screenshot of a cell phone

Description automatically generated

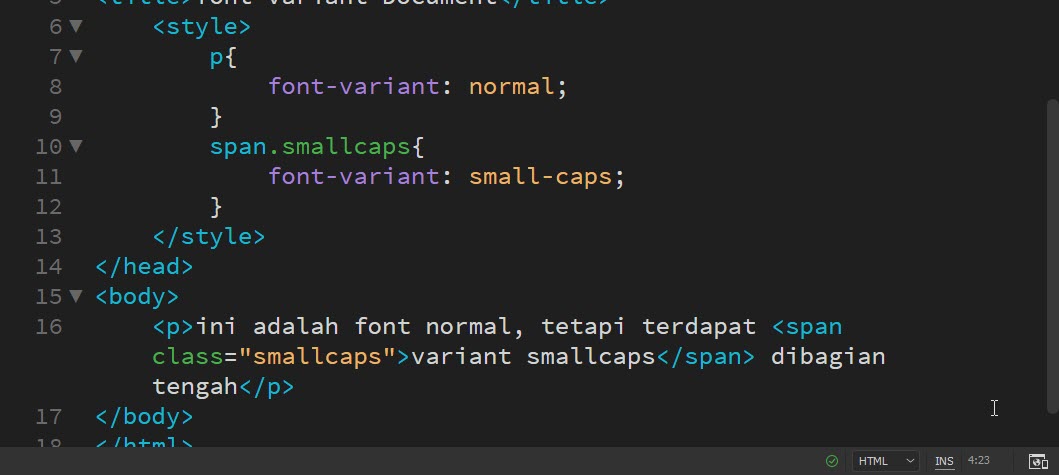
Gambar 31 font-style pada browser.

## Properti font-variant

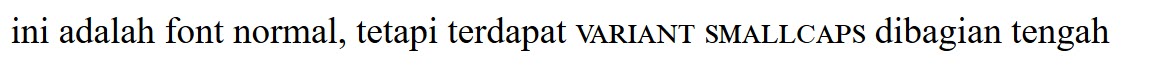
Property ini untuk mengubah bentuk huruf yang tadinya berukuran kecil menjadi semuanya berbentuk huruf capital atau normal.

Terdapat dua nilai pada property ini : **normal** dan **small-caps.**

Berikut contoh penggunaan font-variant pada dokumen html.



Gambar 32 HTML dan CSS font-variant.



Gambar 33 hasil pada browser font-variant.

# Teks

Bab ini melanjutkan kontrol teks pada css di bab sebelumnya. Kontrol teks yang kedua adalah kontrol yang memiliki efek pada teks terlepas dari font yang digunakan. Misalnya, warna teks, jarak antara kata atau huruf dll.

Berikut adalah tabel yang menunjukan property format teks :

Tabel 5 Properti Kontrol Teks CSS.

|  |  |
| --- | --- |
| Properti | Tujuan |
| color | Menentukan warna teks. |
| text-align | Menentukan perataan horizontal teks dalam elemen yang mengandungnya. |
| vertical-align | Menentukan perataan vertical teks dalam elemen yang mengandungnya. |
| text-decoration | Menentukan apakah teks harus diberi underline, overline atau dicoret. |
| text-indent | Menentukan indentasi dari batas kiri untuk teks. |
| text-transform | Menentukan bahwa konten elemen semuanya harus huruf besar, huruf kecil atau kapitalisasi. |
| text-shadow | Menentukan bahwa teks harus memiliki banyangan. |
| letter-spacing | Mengontrol lebar antara huruf. |
| word-spacing | Mengontrol jumlah ruang antara setiap kata. |
| white-space | Menentukan apakah whitespace harus dikecilkan, dipertahankan atau dicegah dari area yang ada. |
| direction | Menentukan arah teks. |

## Properti color

Property ini digunakan untuk mengatur warna pada teks yang dibuat. Nilai warna bisa menggunakan nama secara langsung *(red,blue,yellow),* nilai RGB (*Red,Green, Blue*), nilai hexadecimal, atau HSL (*Hue Saturation Lightness*).

* Untuk penulisan menggunakan format RGB intensitas warna memiliki nilai diantara 0 dan 255 pada setiap parameter (red, green, blue) dan format penulisan adalah **rgb(x,x,x)** dimana x adalah nilai parameter.
* Untuk penulisan menggunakan nilai hexadecimal bagian depan diberi awalan symbol “#” tanpa tanda kutip dengan format #rrggbb dinaman rr (red), gg (green), bb (blue) adalah nilai hexadecimal dimulai dari 00 sampai ff.
* Untuk penulisan menggunakan format hsl memiliki nilai sebagai berikut :
  + Hue adalah derajat pada roda warna dari 0 hingga 360. 0 berwarna merah, 120 berwarna hijau, dan 240 berwarna biru.
  + Saturation adalah nilai persentase penuh tidaknya sebuah warna, 0% berarti warna abu-abu, dan 100% adalah warna penuh.
  + Lightness juga adalah persentase gelap terangnya sebuah warna. 0% adalah hitam, 50% tidak terang atau gelap, 100% putih.

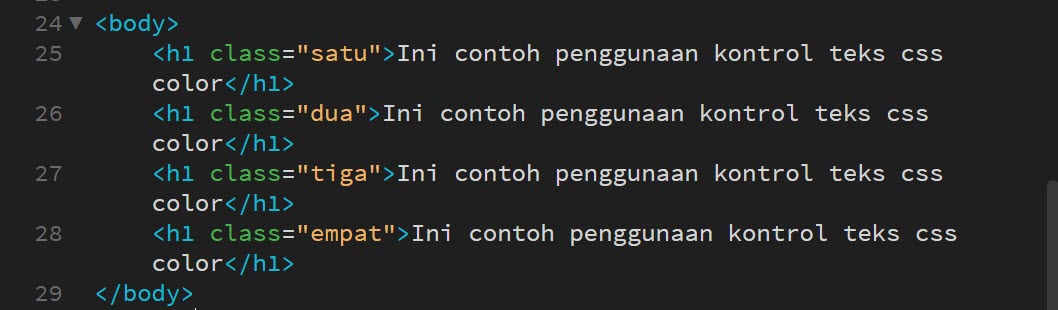
Dengan format penulisan **hsl(h,s,l)**

Berikut adalah contoh penggunaan property color pada dokumen html.

A screenshot of a computer

Description automatically generated

Gambar 34 css kontrol properti color.



Gambar 35 Script HTML kontrol properti color.

A screenshot of a cell phone

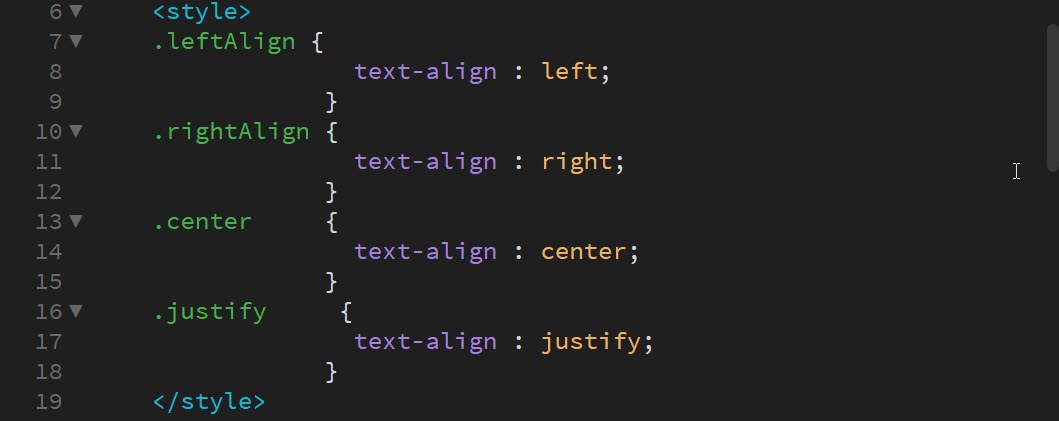
Description automatically generated

Gambar 36 Hasil penggunaan properti color pada browser.

## Properti text-align

Property text-align bekerja seperti atribut pada format HTML. *Align* atau perata yang bisa diatur diantaranya rata kiri (*left*), kanan (*right*), tengah (*center*) dan rata keseluruhan paragraf (*justify*).

Berikut adalah contoh penggunakan property ***text-align*** pada sebuah elemen paragraf dengan ***class*** yang sudah ditentukan pada *script* css.



Gambar 37 Script css class properti text-align.

A screenshot of a cell phone

Description automatically generated

Gambar 38 Hasil pada browser paragraf dengan class sesuai scrip css diatas.

## Properti vertical-align

Properti ini berguna saat bekerja dengan elemen sebaris (*inline*), pada gambar dan bagian teks tertentu. Ini memungkinkan Kita untuk mengontrol posisi vertikal mereka di dalam elemen yang berada didalamnya.

Beberapa nilai yang dapat digunakan pada property ini diantaranya :

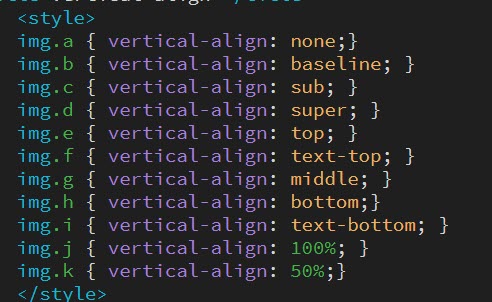
Tabel 6 Nilai Properti vertical-align.

|  |  |
| --- | --- |
| Nilai | Kegunaan |
| baseline | Menyelaraskan dengan garis dasar elemen induk. (Ini adalah pengaturan standar.) |
| sub | Membuat subscript elemen. Dengan gambar, bagian atas gambar harus pada garis dasar. Dengan teks, bagian atas badan font harus pada garis dasar. |
| super | Membuat superscript elemen. Dengan gambar, bagian bawah gambar harus sejajar dengan bagian atas font. Dengan teks, bagian bawah descender (bagian huruf seperti g dan p yang berada di bawah garis teks) harus sejajar dengan bagian atas badan font. |
| top | Sejajarkan bagian atas teks dan bagian atas gambar dengan bagian atas elemen tertinggi di baris. |
| text-top | Sejajarkan bagian atas teks dan bagian atas gambar dengan bagian atas teks tertinggi di baris. |
| middle | Sejajarkan titik tengah vertikal elemen dengan titik tengah vertikal induk. |
| bottom | Sejajarkan bagian bawah teks dan bagian bawah gambar dengan bagian bawah elemen terendah pada baris. |
| text-bottom | Sejajarkan bagian bawah teks dan bagian bawah gambar dengan bagian bawah teks terendah pada baris. |

Properti ini juga dapat menurunkan (inherit) dari nilai dengan persentase.

Berikut adalah contoh penggunaan property vertical-align.

Script css : dengan class berbeda pada setiap nilai properti



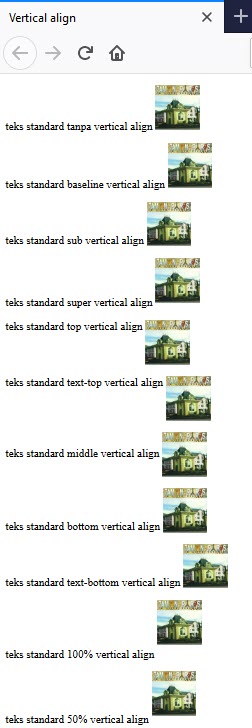
Gambar 39 Script css vertical-align properti.

Script HTML : dengan membuat sebuah teks dan gambar sehingga teks menyesuaikan penyerataan pada sebuah gambar.



Gambar 40 Script HTML Vertical-align.

Hasil pada Browser :



Gambar 41 Vertical-align properti pada browser.

## Properti text-decoration

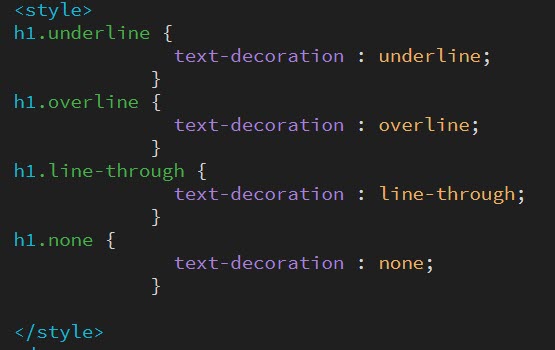
Properti ini digunakan untuk memberikan variasi pada suatu teks atau penanda. Beberapa nilai yang dapat digunakan pada properti ini adalah :

Tabel 7 Nilai properti text-decoration.

|  |  |
| --- | --- |
| nilai | kegunaan |
| underline | Menambahkan baris di bawah konten. |
| overline | Menambahkan baris di atas konten. |
| line-through | Menambahkan garis melalui bagian tengah konten, seperti teks yang dicoret. Secara umum, ini harus digunakan hanya untuk menunjukkan teks yang ditandai untuk dihapus. |
| none | Menghapus semua variasi teks pada suatu elemen. |

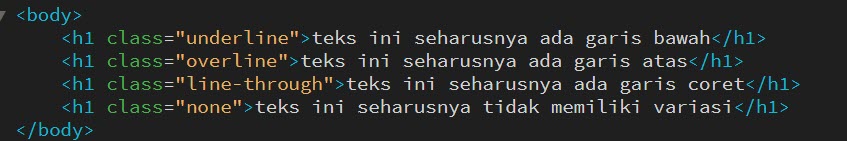
Berikut adalah contoh penggunaan properti text-decoration.

Script css : ditulis pada bagian head



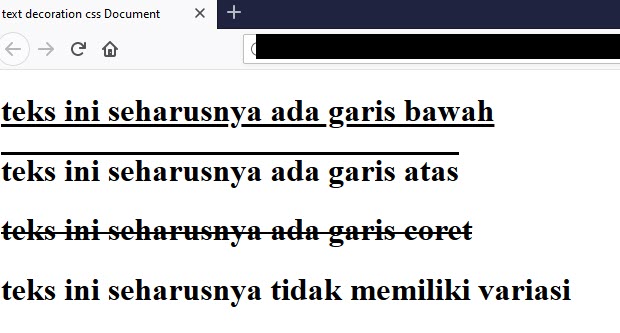
Gambar 42 Text-decoration property.

Script HTML : ditulis pada bagian body



Gambar 43 HTML Text decoration properti.

Hasil pada browser :



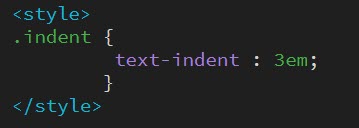
Gambar 44 hasil text-decoration properti pada browser.

## Properti text-indent

Properti ini memungkinkan Kita untuk mengatur indentasi baris teks pertama dalam suatu elemen, sehingga baris pertama pada paragraf tersebut menjorok kedalam sesuai dengan nilai yang sudah Kita tentukan.

Berikut adalah contoh penggunaan properti text-indent :

Script css :



Gambar 45 properti text-indent css.

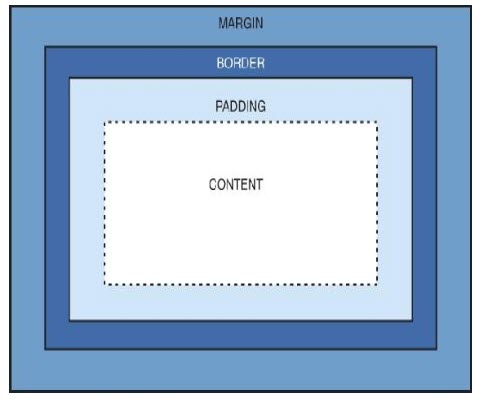
# Box Model

*Box model* adalah konsep penting dalam CSS karena menentukan bagaimana elemen diposisikan dalam jendela browser. Karena CSS memperlakukan setiap elemen seolah-olah berada didalam sebuah kotak (*box*).

Properti box model yang penting dapat dilihat pada tabel dibawah ini :

Tabel 10 Properti box model.

|  |  |
| --- | --- |
| Properti | Deskripsi |
| border | Bahkan jika Anda tidak dapat melihatnya, setiap kotak memiliki batas (*border*). Ini memisahkan tepi satu kotak dari kotak lain di seKitarnya. |
| margin | Margin adalah jarak antara batas kotak dan kotak di sebelahnya. |
| padding | Padding adalah ruang antara isi kotak dan perbatasannya |



Gambar 66 Setiap Elemen HTML Direpresentasikan CSS menggunakan box model.

Kita dapat menggunakan CSS untuk secara individual mengontrol *border, margin,* dan *padding* di setiap sisi kotak. Kita dapat menentukan lebar, gaya garis, dan warna yang berbeda untuk setiap sisi batas kotak.

Sifat *padding* dan *margin* sangat penting dalam menciptakan apa yang oleh perancang disebut sebagai ruang putih (*white space),* ini adalah ruang di antara berbagai bagian halaman.

Misalnya, jika Kita memiliki kotak dengan batas hitam dan kotak berisi teks hitam, Kita tidak ingin teks menyentuh perbatasan karena akan membuat teks lebih sulit dibaca. Memberikan *padding* kotak membantu memisahkan teks dari garis di seKitar tepi.

Sementara itu, anggaplah Kita memiliki dua kotak di samping satu sama lain, keduanya berbatasan. Jika tidak ada margin di antara mereka, kotak akan bertemu satu sama lain, dan garis tempat kotak bertemu bisa terlihat lebih tebal daripada garis lainnya.

## Ilustrasi box model

Untuk mengilustrasikan *box model*, Kita bisa menambahkan *border* pada setiap elemen dalam halaman web.

Script css : setiap elemen memiliki border box masing-masing dan pada element <h1> dan <b> diberi warna abu-abu untuk membantu membedakan dari elemen lain.

A screenshot of text

Description automatically generated

Gambar 67 box elemen css.

Script HTML :

A screenshot of a cell phone

Description automatically generated

Gambar 68 box elemen html.

Hasil pada browser :

A screenshot of a social media post

Description automatically generated

Gambar 69 Box elemen pada browser.

Pada buku lain yang membahas dasar HTML (Panduan Definitif untuk HTML5) dibahas mengenai perbedaan antara elemen blok dan elemen *inline*, perbedaan ini menjadi sangat penting ketika bekerja dengan CSS karena ia menentukan bagaimana setiap kotak ditampilkan.

# Border

Properti Box model yang pertama adalah border. Berbeda dengan border pada HTML, border dalam CSS berdiri sendiri tanpa bantuan tabel. Salah satu kelebihan border salam CSS adalah Kita bisa memilih *style* yang beragam, selain itu Kita bisa mengatur bagian atas, bawah, kanan, dan kiri border dengan pengaturan yang berbeda.

## Properti Border

Properti border memungkinkan Kita menentukan bagaimana batas kotak yang mewakili elemen akan terlihat. Ada 3 (tiga) properti border yang dapat diubah :

1. ***border-color*** digunakan untuk mengatur warna pada border.
2. ***border-style*** digunakan untuk mengatur hiasan pada border dapat berupa *solid, dashed, double line atau* salah satu dari nilai yang mungkin lainnya.
3. ***border-width*** digunakan untuk mengatur lebar sebuah border.

## Properti border-width

Properti ini memungkinkan Kita untuk menentukan lebar dari sebuah border, biasanya ditentukan dalam satuan *pixel* (px). Nilai properti border-width tidak dapat menggunakan persentase, meskipun Kita dapat menggunakan absolut unit atau relative unit, atau salah satu dari nilai berikut :

* ***thin***
* ***medium***
* ***tick***

Nilai diatas tidak ditentukan dalam rekomendasi css, sehinggal lebar sebenarnya yang sesuai dengan nilai diatas ini bergantung pada browser.

Kita dapat mengubah lebar batas bagian bawah, kiri, atas, dan kanan kotak secara individual menggunakan properti berikut:

* border-bottom-width
* border-right-width
* border-top-width
* border-left-width

## Properti *border-style*

Properti ini memungkinkan Kita untuk menentukan garis apa yang akan digunakan sebagai pembatas (*border*). Nilai *default* dari properti ini adalah “***none****”*jadi tidak ada border yang akan ditampilkan secara otomatis. Berikut adalah nilai yang dapat digunakan pada *border-style.*

Tabel 11 Properti border-style

|  |  |
| --- | --- |
| Nilai | Deskripsi |
| *none* | Tidak ada border (setara dengan ***border-width***:0;) |
| *solid* | Border adalah garis solid tunggal. |
| *dotted* | Border adalah serangkaian titik. |
| *dashed* | Border adalah serangkaian garis pendek. |
| *double* | Border adalah dua garis solid, nilai properti ***border-width*** menciptakan jumlah dari dua garis dan ruang diantara mereka. |
| *groove* | Border tampak seolah-olah diukir kehalaman. |
| *ridge* | Border terlihat kebalikan dari ***groove.*** |
| *inset* | Border membuat kotak terlihat seperti tertanam di *page.* |
| *outset* | Border membuat kotak tampak seperti keluar kanvas. |
| *hidden* | Sama seperti ***none,*** kecuali dalam hal resolusi konflik untuk elemen tabel. |

Kita dapat mengubah *style* batas bagian bawah, kiri, atas, dan kanan kotak secara individual menggunakan properti berikut:

* border-bottom-style
* border-right- style
* border-top- style
* border-left- style

## Properti *border-color*

Properti ini memungkinkan Kita untuk mengubah warna border diseKitar kotak.

Sintaks :

Elemen { border-color : nilai valid warna; }

Nilai dapat digunakan menggunakan nilai valid warna pada css (hexadecimal, rgb, rgba,hsl,hsla, nama warna).

Kita dapat mengubah warna batas bagian bawah, kiri, atas, dan kanan kotak secara individual menggunakan properti berikut:

* border-bottom-color
* border-right-color
* border-top-color
* border-left-color

## Properti *border (singkat)*

Properti border memungkinkan Kita secara singkat menentukan warna, style dan lebar dalam satu properti.

Jika Kita menggunakan singkatan ini, nilainya tidak boleh memiliki apa pun (selain spasi) di antara mereka.

Contoh :

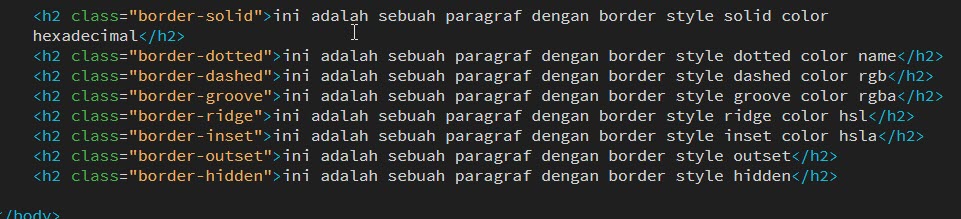
p { border : 4px solid red; }

Kita juga dapat menentukan warna, style, dan lebar garis secara terpisah untuk setiap sisi kotak dengan cara yang sama menggunakan properti ini:

* border-bottom
* border-top
* border-left
* border-right

berikut adalah contoh penggunaan properti border :

script html :



Gambar 70 Script HTML Properti border.

Script css dibagi menjadi 2 (dua) gambar yang dituliskan pada bagian ***head html***:

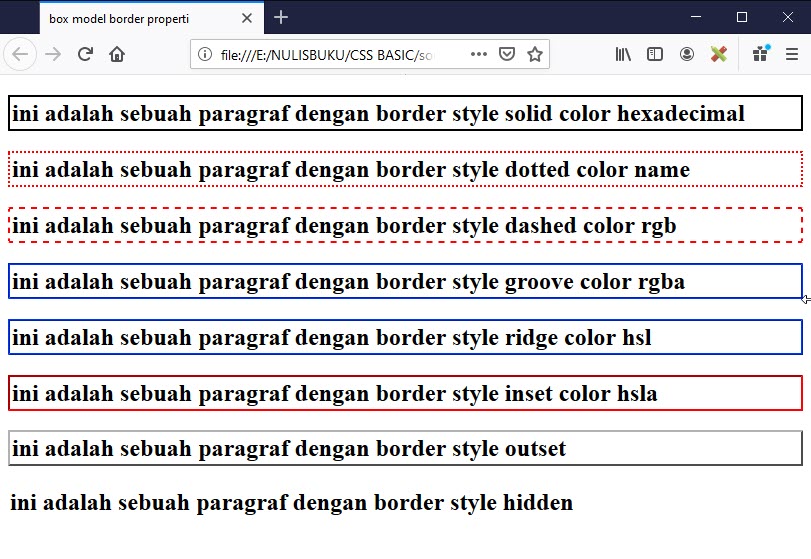


Gambar 71 Script css border bagian 1.



Gambar 72 Script css border bagian 2.

Hasil pada browser :



Gambar 73 Hasil css border pada browser.

# *Margin* dan *Padding*

Pengaturan *margin* dan *padding* yang baik dapat membuat dokumen HTML terlihat lebih rapih dan profesional. Dengan menggunakan CSS Kita bisa mengaturnya sesuai dengan keinginan.

## Properti *Padding*

Properti padding memungkinkan Kita menentukan berapa banyak ruang yang harus muncul antara konten elemen dan border.

Nilai properti ini paling sering ditentukan dalam piksel. Meskipun, itu dapat menggunakan salah satu unit panjang yang Kita temui sebelumnya, persentase, atau kata yang diwarisi (*inherit*).

*Padding* suatu elemen tidak mewarisi secara *default*, jadi jika elemen ***<body>*** memiliki properti *padding* dengan nilai 50 piksel, ini tidak secara otomatis berlaku untuk semua elemen lain di dalamnya. Jika nilai *inherit* diterapkan ke elemen apa pun, hanya dengan demikian mereka dapat memiliki *padding* yang sama dengan elemen induknya.

Jika persentase digunakan, persentase kotak berisi, dan jika nilai 10 persen ditentukan, akan ada 5 persen dari masing-masing sisi kotak sebagai *padding*.

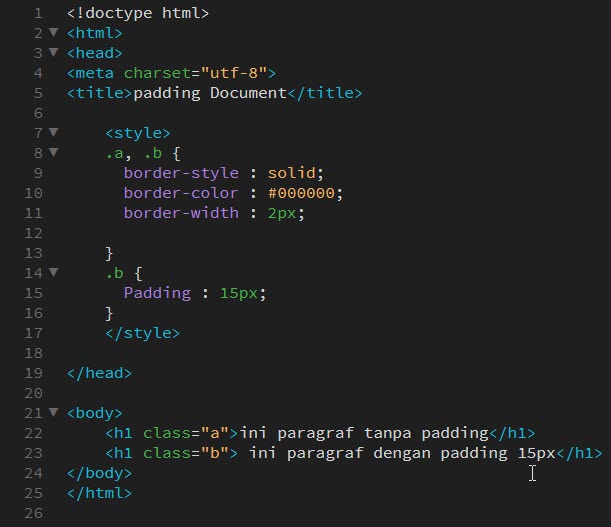
Kita dapat menentukan jumlah *padding* yang berbeda di dalam setiap sisi kotak menggunakan properti berikut:

* ***padding-bottom***
* ***padding-top***
* ***padding-left***
* ***padding-right***

Atribut *padding* sangat membantu untuk membuat ruang putih (*white space*) antara konten elemen dan setiap perbatasan yang dimilikinya.

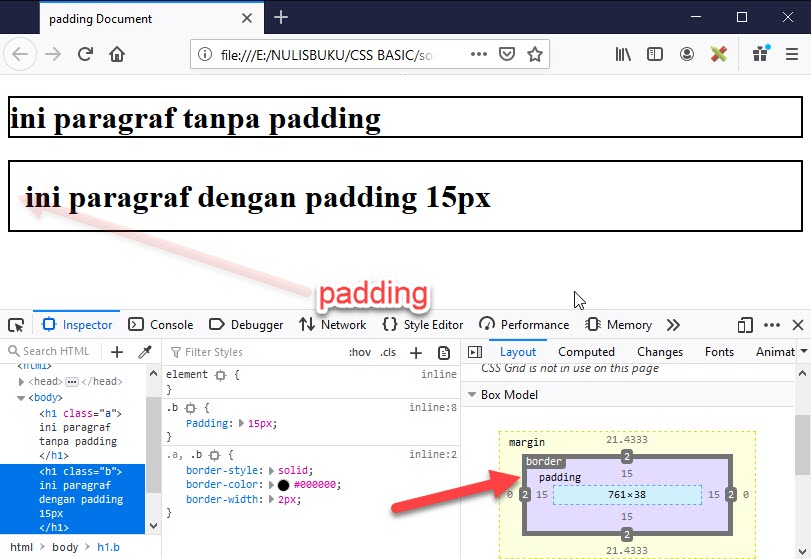
Berikut adalah contoh penggunaan properti padding.

Script html dan css :



Gambar 74 Script HTML dan CSS properti Padding,

Hasil pada browser dengan *inspect element* :



Gambar 75 Hasil Properti padding pada browser.

Terkadang, suatu elemen mungkin tidak memiliki *border* yang terlihat, tetapi memiliki warna atau pola latar belakang. Dalam kasus seperti itu, berikan *padding* pada beberapa kotak dapat membantu membuat desain Kita lebih menarik.

## Properti *margin*

Properti margin mengontrol celah antara kotak, dan nilainya panjang menggunakan angka, persentase, atau warisan (*inherit*), yang masing-masing memiliki arti yang persis sama seperti yang terjadi pada properti *padding* yang baru saja Kita lihat.

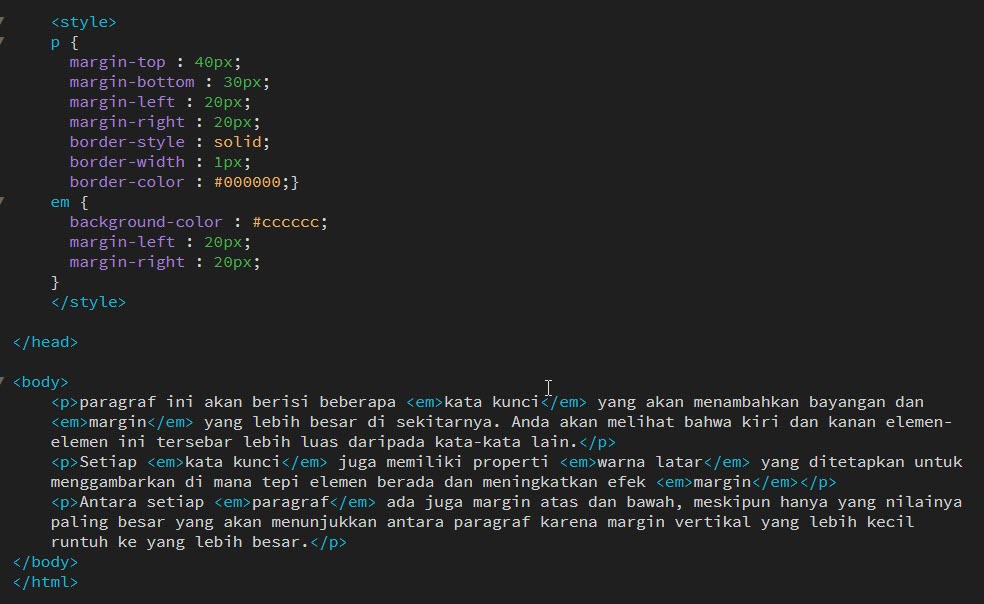
Seperti halnya properti *padding*, nilai properti *margin* tidak diwarisi oleh elemen turunan kecuali Anda menggunakan nilai inherit.

Juga, ingatlah bahwa ketika satu kotak berada di atas kotak yang lain, hanya yang lebih besar dari dua margin yang ditampilkan (atau jika keduanya sama, ukuran satu margin).

Kita juga dapat mengatur nilai yang berbeda untuk margin di setiap sisi kotak menggunakan properti berikut:

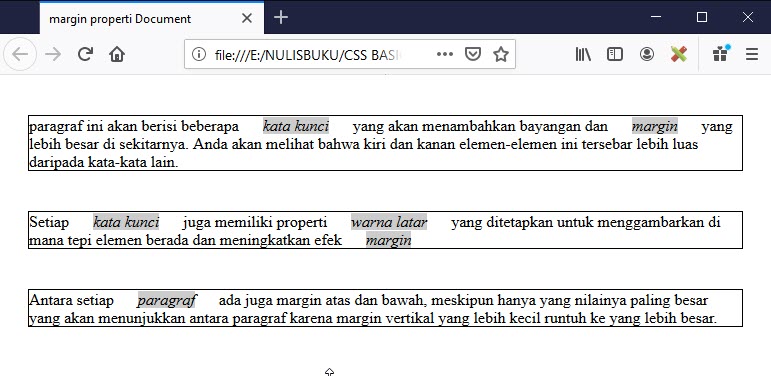
* ***margin-bottom***
* ***margin-top***
* ***margin-left***
* ***margin-right***

Berikut adalah contoh penggunaan properti margin pada html dan css:



Gambar 76 Script html dan css properti margin.

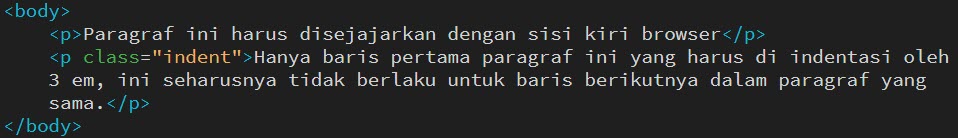
Hasil pada browser :



Gambar 77 Hasil pada browser properti margin.

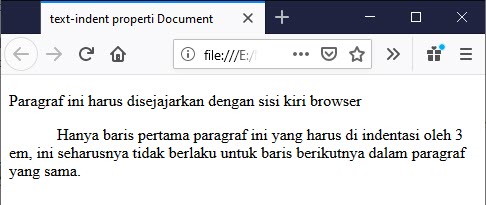
Kita dapat melihat tiga paragraf, yang terlihat seolah-olah diberi spasi sama. Namun, mereka memiliki *margin* yang lebih tinggi di bagian atas daripada bagian bawah, dan oleh karena itu di mana dua kotak bertemu, margin bawah diabaikan — marginnya diciutkan. (Ini hanya terjadi pada **margin vertikal**, bukan margin kiri dan kanan.)

Script HTML :



Gambar 46 HTML Text indent properti.

Hasil pada browser :



Gambar 47 hasil text indent pada browser.

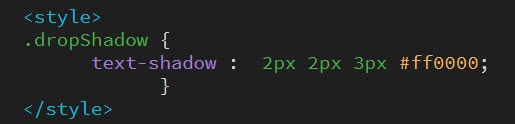
## Properti text-shadow

Property ini memberikan efek bayangan pada sebuah teks atau paragraf. Sering digunakan di media cetak. Nilai untuk propreti ini cukup rumit karena dapat mengambil warna diikuti 3 (tiga) nilai yang lainnya.

Dua nilai yang pertama menentukan seberapa jauh bayangan dari teks asli menggunakan koordinat X dan Y, nilai yang ketiga sebelum nillai warna adalah menentukan seberapa blur banyangan seharusnya, nilai terakhir adalah nilai warna pada umumnya sebuah css.

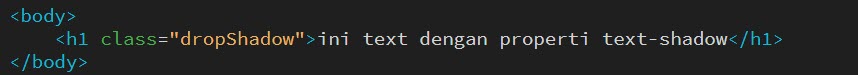
Contoh penggunaan properti text-shadow adalah sebagai berikut :

Script css : pada bagian head dibuat tag style dengan class dropShadow. Nilai koordinat x dan y adalah 2px dan nilai blur 3px dengan warna merah (#ff0000).



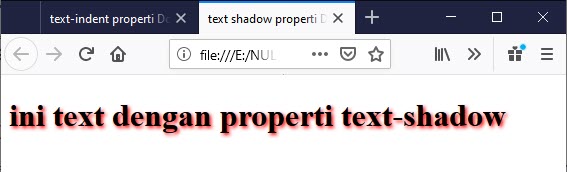
Gambar 48 css text-shadow properti.

Script HTML :



Gambar 49 HTML Text shadow properti.

Hasil pada browser :



Gambar 50 browser text shadow properti.

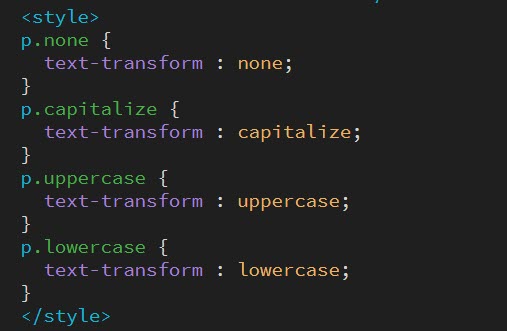
## Properti text-transform

Properti ini digunakan untuk mengubah teks pada suatu konten elemen. Baik menjadi *capitalize, uppercase* atau *lowercase*. Beberapa nilai yang dapat digunakan pada property ini adalah sebagai berikut :

|  |  |
| --- | --- |
| Nilai | Kegunaan |
| none | tidak ada perubahan yang terjadi |
| capitalize | mengubah karakter huruf pertama menjadi kapital / hurus besar. |
| uppercase | membuat seluruh konten elemen huruf besar. |
| lowercase | membuat seluruh konten elemen huruf kecil. |

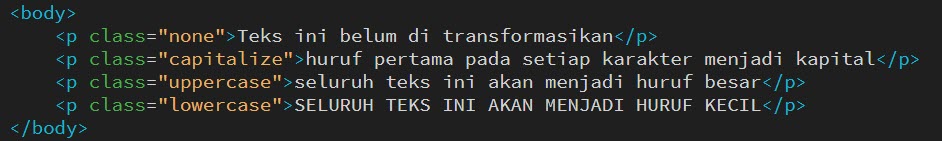
Berikut adalah contoh penggunaan *text-decoration* properti.

Script css : pada script ini Kita akan membuat class dengan fungsi yang sama dengan nilai properti text-decoration.



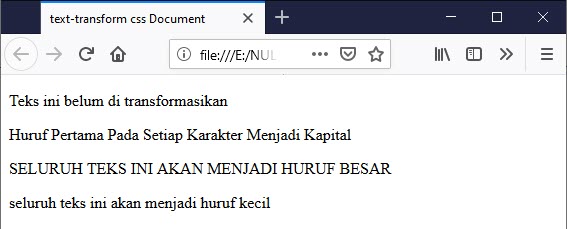
Gambar 51 css text-transform properti.

Script HTML : Kita akan membuat beberapa paragraf elemen dengan class yang sudah dibuat pada script css.



Gambar 52 html file text-transform properti.

Hasil : hasil pada browser dapat dilihat pada gambar dibawah ini.



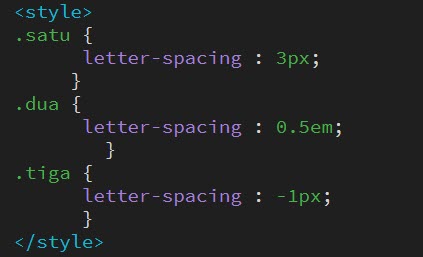
Gambar 53 Browser text-transform.

## Properti letter-spacing

Properti ini memungkinkan Anda untuk mengatur jarak antar karakter. Hal ini biasanya digunakan oleh seorang desainer cetak untuk mengatur jarak antar karakter tersebut. Jika Kita ingin menambah atau mengurangi jarak antar karakter, kemungkinan besar Kita menetukannya dalam **pixel** atau sesuatu yang dikenal dengan **em**. Meskipun satuan panjang lain yang didukung oleh CSS bisa digunakan).

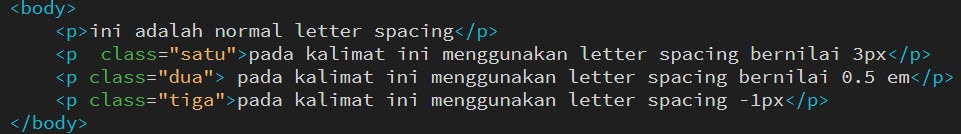
Berikut adalah contoh penggunaan properti letter-spacing.

Script css : Saya akan menggunakan 3 buah class untuk memberi nilai pada properti letter-spacing.



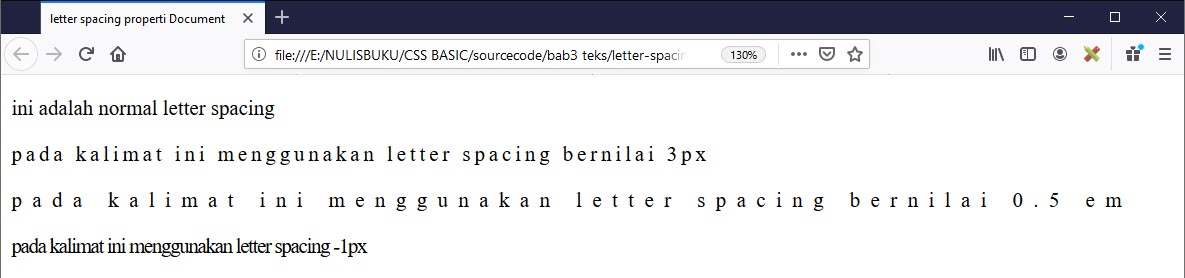
Gambar 54 letter spacing properti css.

Script html : memiliki 3 paragraf dengan class yang sudah dibuat pada script css.



Gambar 55 letter spacing html.

Hasil pada browser :

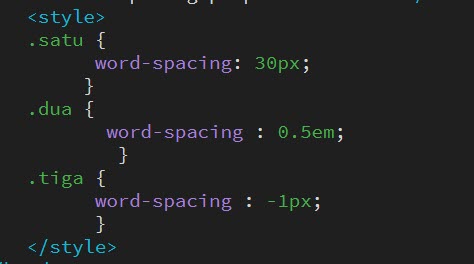


Gambar 56 letter spacing browser.

## Properti word-spacing

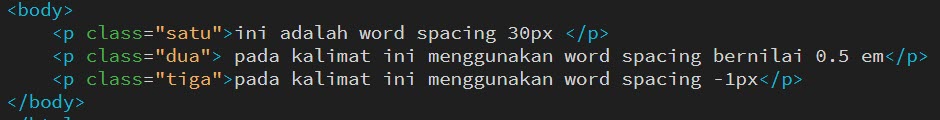
Pada properti ini Kita dapat mengatur jarak antar kata pada sebuah dokumen HTML. Penggunaannya sama seperti letter-spacing properti. Berikut adalah contoh penggunaan properti word-spacing.

Script css :



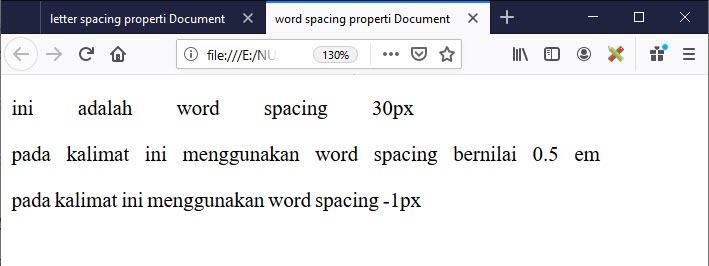
Gambar 57 word-spacing properti css.

Script HTML



Gambar 58 word-spacing HTML.

Hasil pada browser :



Gambar 59 word-spacing browser.

## Properti white-space

Properti ini mengontrol apakah white space (ruang putih pada editor) dipertahankan atau diabaikan. Ada beberapa nilai yang dapat digunakan pada properti white-space terkait penggunaan white space pada sebuah dokumen html.

Tabel 8 Properti white space.

|  |  |
| --- | --- |
| nilai | kegunaan |
| normal | normal sesuai halaman web. |
| pre | mempertahankan *white space* seperti pada editor HTML. |
| pre-line | mempertahakan white space seperti pada editor HTML tetapi pada value pre-line jarak setiap per-kata hanya akan menampilkan satu spasi seperti pada umumnya walaupun anda mengetikkan lebih dari satu spasi.dapat membuat baris baru jika diperlukan. |
| pre-wrap | mempertahakan white space seperti pada editor apabila baris teks akan mengalir keluar melebihi batas elemen ia akan membuat baris baru sehingga baris teks tetap berada di dalam area pembungkusnya. |
| nowarp | teks akan terus memanjang (tidak membuat baris baru untuk menyesuaikan tampilan di browser) jika tidak secara eksplisit menggunakan elemen <br>. |

Berikut adalah contoh penggunaan properti white-space.

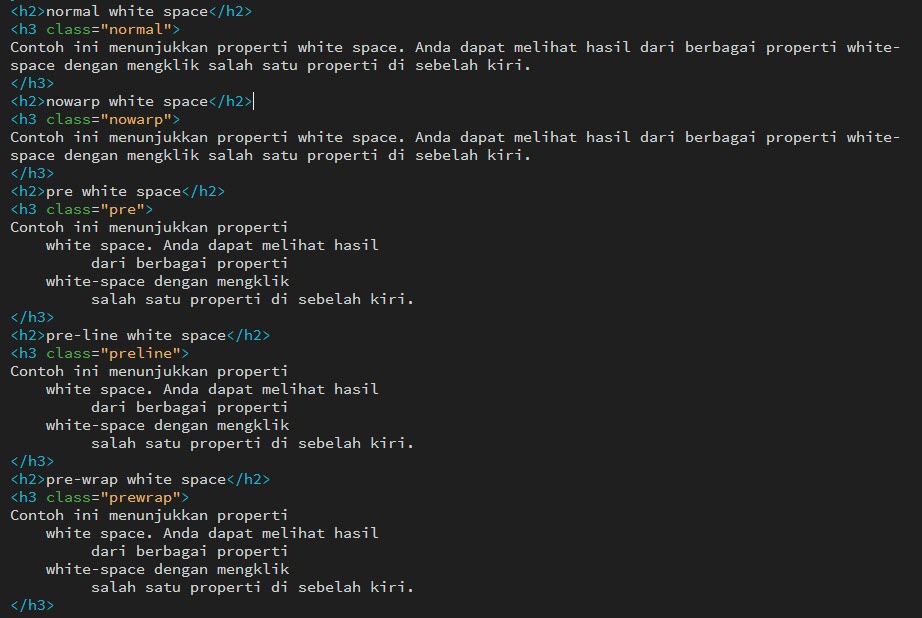
Script css : akan dibuat sebuah class dengan nilai properti white-space yang berbeda-beda.

A close up of text on a black background

Description automatically generated

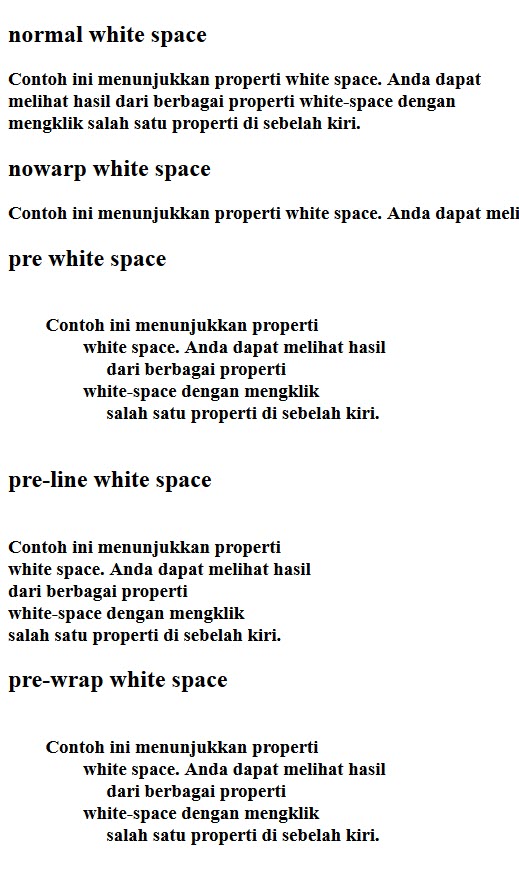
Gambar 60 white space properti.

Script HTML :



Gambar 61 white space properti html.

Hasil pada browser :



Gambar 62 white space properti pada browser.

## Properti direction

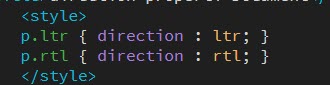
Properti ini seperti elemen <dir> pada HTML yaitu menentukan arah dinama teks harus mengalir. Berikut adalah properti dari *direction* :

Tabel 9 direction properti.

|  |  |
| --- | --- |
| nilai | kegunaan |
| ltr | teks mengalir dari kiri ke kanan. |
| rtl | teks mengalir dari kanan ke kiri. |
| inherit | teks mengalir sesuai dengan arah dari elemen induk. |

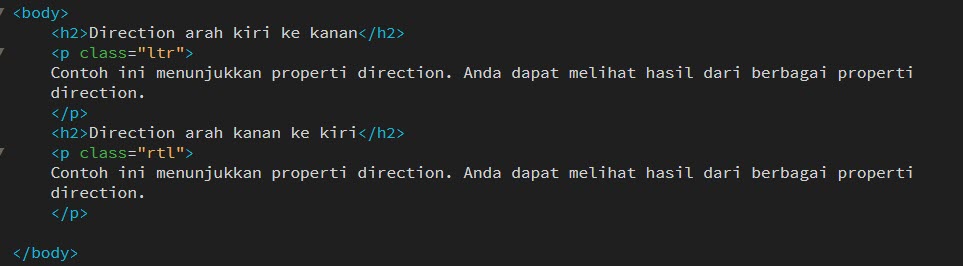
Berikut adalah contoh penggunaan properti direction :

Script CSS:



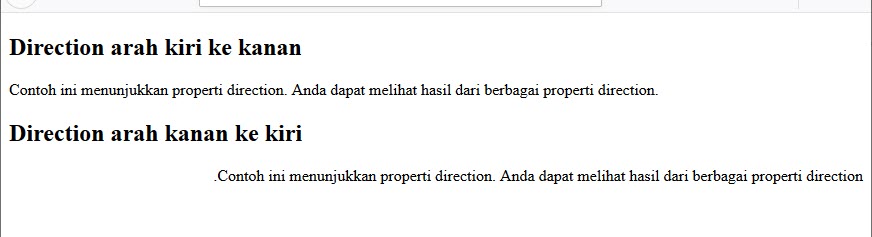
Gambar 63 properti direction css.

Script HTML :



Gambar 64 properti direction html.

Hasil pada browser :



Gambar 65 hasil pada browser properti direction.

# CSS Layout

Properti **display** adalah properti CSS paling penting untuk mengontrol tata letak.

Properti **display** menentukan jika / bagaimana sebuah elemen ditampilkan.

Setiap elemen HTML memiliki nilai tampilan default tergantung pada jenis elemennya. Nilai tampilan default untuk sebagian besar elemen adalah **blok** atau **inline**.

## Block Level Elements

*Block Level Elements* selalu dimulai pada baris baru dan menempati lebar penuh yang tersedia (membentang ke kiri dan kanan sejauh mungkin).

Contoh :

* <div>
* <h1> - <h6>
* <p>
* <form>
* <header>
* <footer>
* <section>

## Inline Elements

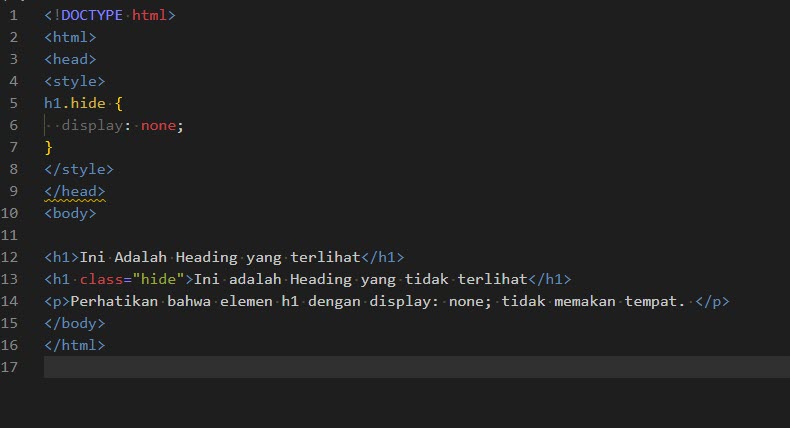
Inline Elements tidak dimulai pada baris baru dan hanya membutuhkan lebar yang diperlukan.

Contoh :

* <span>
* <a>
* <img>

## Properti Display : none;

biasanya digunakan dengan JavaScript untuk menyembunyikan dan menampilkan elemen tanpa menghapus dan membuatnya kembali.



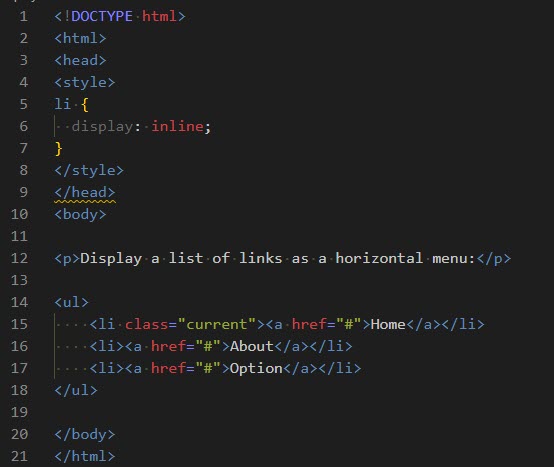


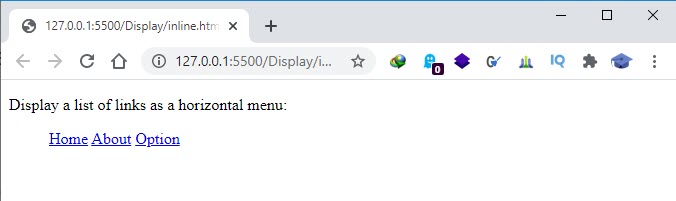
## Override The Default Display Value

Seperti yang disebutkan, setiap elemen memiliki nilai tampilan default. Namun, Anda dapat menimpanya.

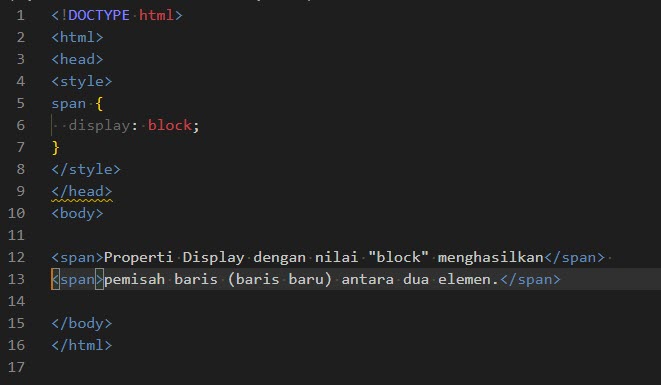
Mengubah elemen inline menjadi elemen blok, atau sebaliknya, dapat berguna untuk membuat laman terlihat dengan cara tertentu, dan tetap mengikuti standar web.

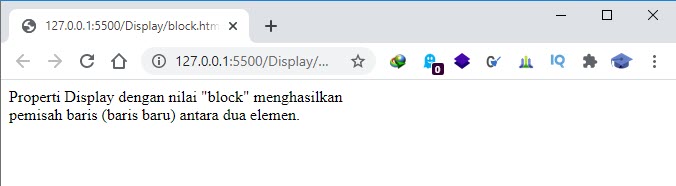
Contoh umum adalah membuat elemen inline <li> untuk menu horizontal:





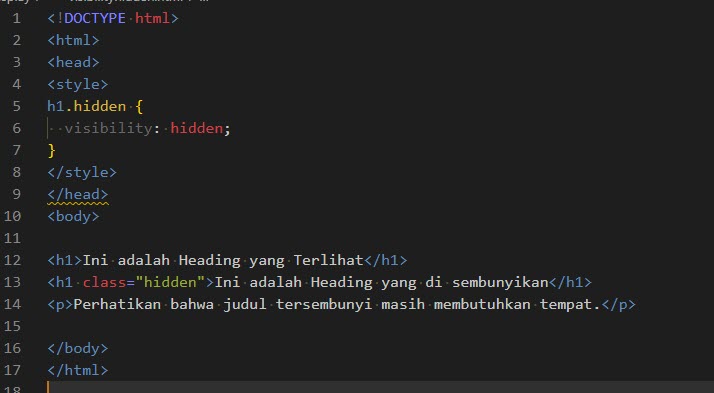
Contoh : Inline menjadi blok





## Properti visibility : hidden;

Selain dengan menggunakan display : none; ada properti css yang berfungsi untuk menyembunyikan elemen yaitu visibility : hidden. Namun, elemen tersebut akan tetap menempati ruang yang sama seperti sebelumnya. Elemen tersebut akan disembunyikan, tetapi masih memengaruhi tata letak:





# CSS Layout Float & Clear

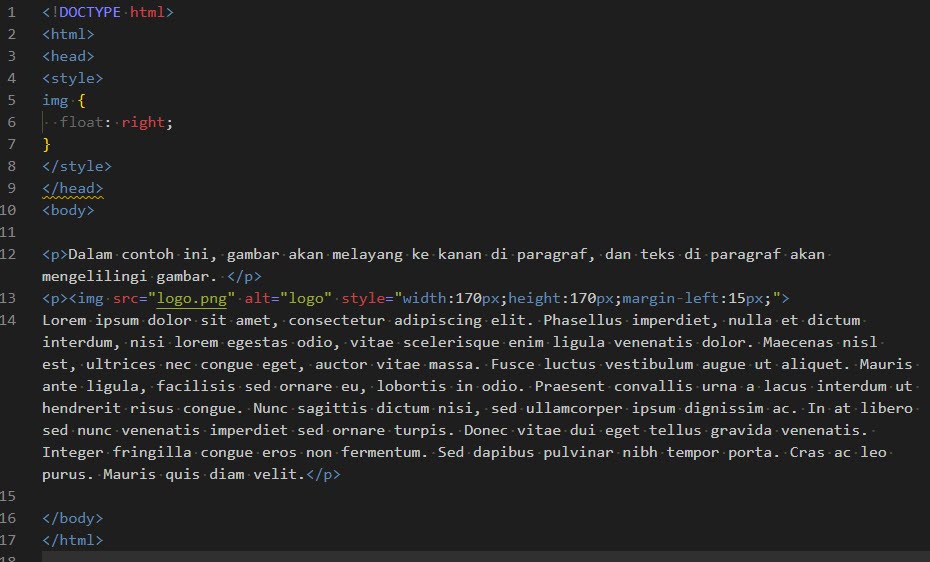
Jika Kita sering melakukan penempatan gambar pada sebuah teks dalam ms word maka terkadang sebuah gambar ditempatkan seolah - oleh berada pada lapisan atas sebuah paragraf tertentu.

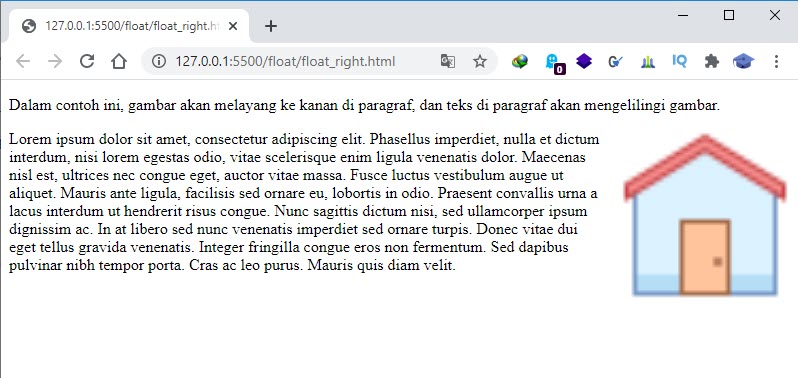
Properti CSS float menentukan bagaimana sebuah elemen harus mengapung (float pada layout tertentu) sehingga tampilan seolah - olah berada pada lapisan atasnya.

Properti CSS clear menentukan elemen apa yang dapat mengapung (float) di samping elemen yang dibersihkan dan di sisi mana.

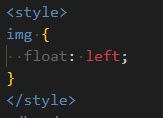
## Properti FLoat

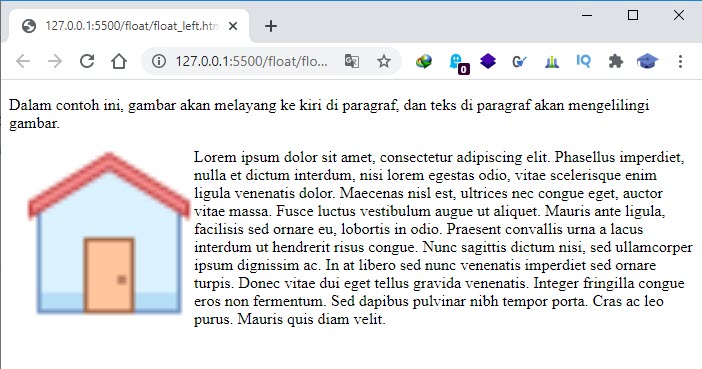
Contoh Float



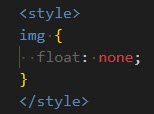


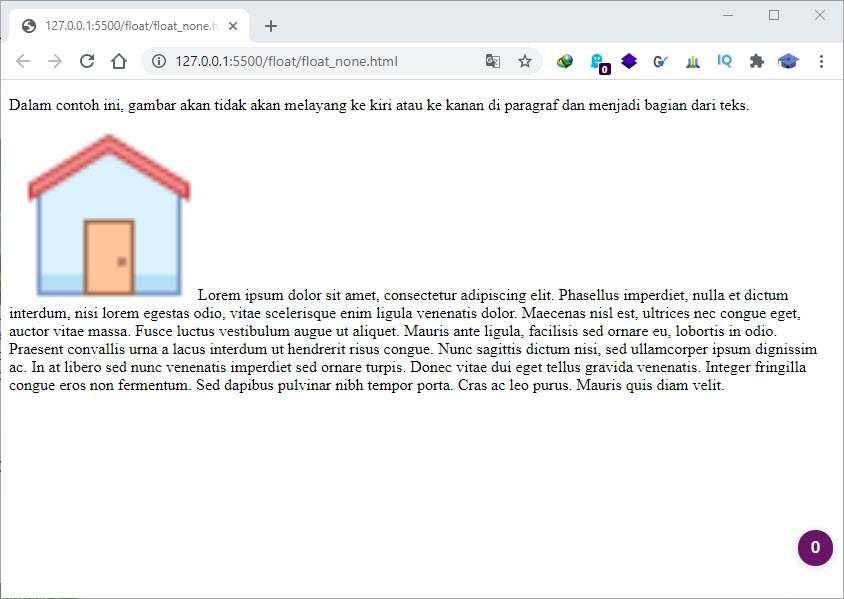
Float left





Float : none





## Properti Clear

Properti menentukan elemen apa yang bisa mengapung di samping elemen yang clear kan dan di sisi mana.

Properti dapat memiliki salah satu nilai berikut:

* none - Memungkinkan elemen mengambang di kedua sisi. Ini default
* kiri - Elemen mengambang tidak diperbolehkan di sisi kiri
* kanan- Tidak ada elemen mengambang yang diizinkan di sisi kanan
* keduanya - Tidak ada elemen mengambang yang diizinkan di sisi kiri atau kanan
* inherit - Elemen mewarisi nilai yang jelas dari induknya

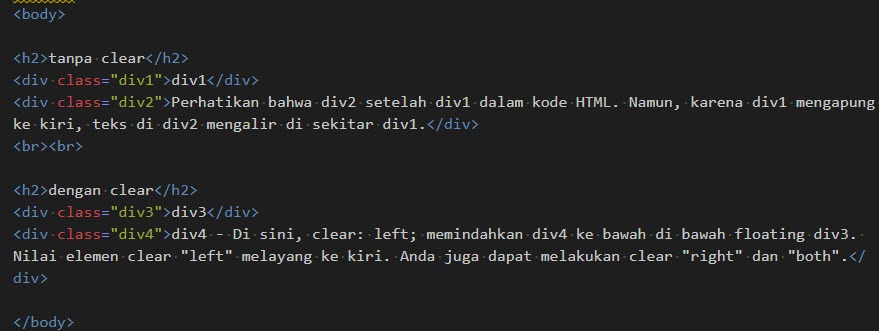
Cara paling umum untuk menggunakan properti clear adalah setelah Anda menggunakan properti float pada sebuah elemen.

Saat membersihkan float, Anda harus mencocokkan yang jelas dengan floating-nya: Jika elemen di float kan ke kiri, maka Anda harus jelas ke kiri. Elemen float Anda akan terus mengambang, tetapi elemen yang telah clear-kan akan muncul di bawahnya pada halaman web.

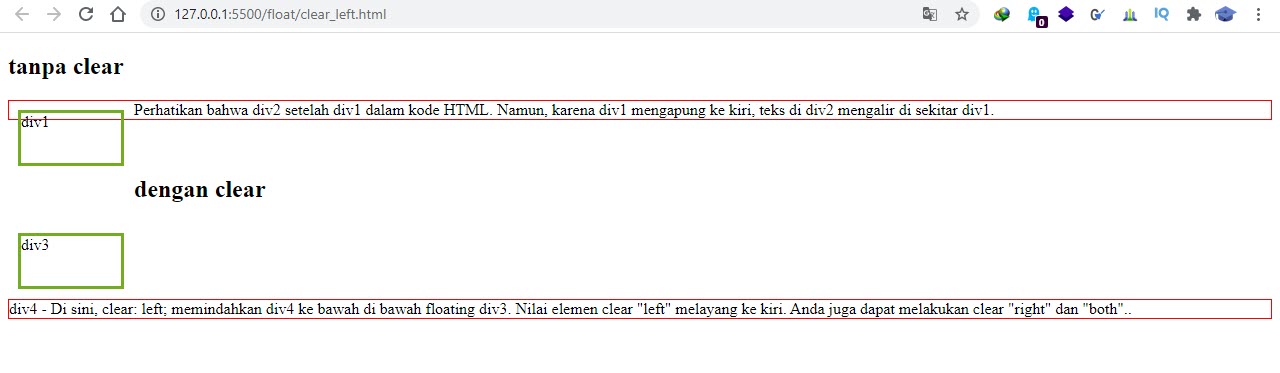
Contoh berikut membersihkan float ke kiri. Berarti tidak ada elemen mengambang yang diizinkan di sisi kiri (div):



html body :



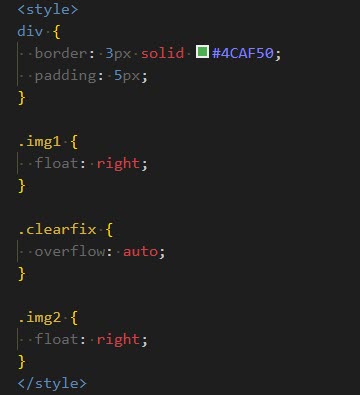
Hasil :



Jika sebuah elemen lebih tinggi dari elemen yang memuatnya, dan itu floatingkan, itu akan "overflow" di luar penampungnya:

Kemudian kita bisa menambahkan **overflow: auto**; ke elemen yang mengandung untuk memperbaiki masalah ini:

style css



body html

