**Formatting Text (Text and Font Styling)**

Setelah kita tahu bagaimana menerapkan CSS pada berkas HTML, mengetahui struktur penulisan CSS, dan menggunakan *selector* sesuai dengan kebutuhan, apa berikutnya? Kita akan fokus mempelajari secara detail bagaimana menerapkan *styling* pada sebuah teks.

Perlu kita ketahui, CSS memisahkan *styling* untuk font dan teks. Kita harus tahu kapan seharusnya menetapkan styling pada font atau menetapkannya pada teks. Mengapa kedua hal tersebut dipisah? Padahal teks dan font merupakan satu kesatuan, di mana setiap teks pasti menggunakan font untuk menampilkannya, bukan? Sebabnya, CSS sendiri mempunyai properti untuk kedua hal tersebut secara masing-masing

Jika kita ingin menetapkan *styling* pada tampilan teks itu sendiri, gunakanlah properti font. Contohnya properti font dapat mengatur tipe font, ukuran, ketebalan, dan lainnya. Sedangkan properti teks digunakan untuk mendukung hal lainnya dalam menampilkan teks seperti, menetapkan *text alignment*, dekorasi, *spacing*, dan masih banyak lagi. Karena dua hal ini dipisah, mari kita bahas satu persatu.

1. **Font Styling**

Ketika kita membuat sebuah dokumen teks, termasuk dokumen cetak, langkah awal kita biasanya adalah menentukan jenis font yang akan digunakan. Pada pengembangan website pun demikian. Dalam CSS, font ditentukan dengan menggunakan beberapa paket properti font. Kita bisa atur tipe font, ukuran, ketebalan, dan gaya. Berikut ini merupakan properti font yang akan kita pelajari antara lain:

* font-family : Menetapkan jenis font yang akan diterapkan pada target.
* font-size : Menentukan ukuran pada teks.
* font-weight : Menentukan ketebalan pada teks.
* font-style : Menetapkan styling yang diterapkan pada teks.
* font-variant : Menentukan teks untuk menggunakan gaya *small cap*s (huruf kapital kecil).
* font : Shorthand dari properti font yang ada.

Mari kita bahas properti tersebut satu persatu secara mendetail.

**font-family**

Pada sub-modul pengenalan CSS kita sudah mencoba menggunakan *font properties* ini untuk mengubah standar font yang ditampilkan pada browser dengan menggunakan font-family pada elemen <body>.

Untuk menuliskan lebih dari satu nilai font, berikut aturan yang harus kita perhatikan:



* Seluruh nilai *font*yang bukan merupakan *generic font families*harus dituliskan secara kapital. Contohnya “Arial” bukan dituliskan “arial”.
* Gunakan tanda koma (,) untuk memisahkan antar nilai *font*yang digunakan.
* Selalu tanda kutip (“) untuk membungkus nilai font yang memiliki spasi pada namanya (Contohnya “*Open Sans*”).

Mungkin kita bertanya-tanya mengapa perlu memberikan lebih dari satu nilai pada *font-family*? Hal tersebut penting karena tidak semua browser mendukung semua jenis font. Memberikan lebih dari satu nilai font dapat menawarkan alternatif tampilan font pada browser. Terutama jika font utama yang diterapkan tidak didukung oleh browser yang digunakan.

Bagaimana urutan prioritasnya? Mulai dari jenis *font*yang pertama dituliskan. Jika *font*pertama didukung oleh browser, maka browser akan menggunakannya. Namun jika tidak, maka browser akan memeriksa nilai *font*yang kedua dan menggunakannya (jika didukung), demikian dan seterusnya.

Pastikan untuk menggunakan [*generic font families*](https://developer.mozilla.org/en-US/docs/Web/CSS/font-family#values) pada akhir nilai properti font-family, karena nilai ini dipastikan didukung oleh seluruh browser saat ini. Lantas apa saja nilai dari generic font families ini? Berikut nilai-nilai generic font families yang dapat kita gunakan untuk fallback mechanism:

* *serif* : jenis font yang memiliki runcing pada garis akhir karakternya. Times New Roman merupakan salah satu jenis serif font.
* *sans-serif* : jenis font yang tidak meruncing pada garis akhir karakternya. Contohnya “Open Sans”, “Fira Sans” dan lainnya.
* *monospace* : jenis font yang memiliki nilai lebar tiap karakternya sama. Consolas merupakan salah satu jenisnya.
* *cursive*: jenis font yang tampak seperti handwriting atau hasil tulisan tangan.
* *fantasy* : jenis font yang merepresentasikan karakteristik yang menyenangkan.
* *system-ui* : jika menerapkan nilai ini maka font yang diterapkan akan sama seperti font yang digunakan pada sistem operasi kita.
* *math* : jenis font yang digunakan untuk penulisan rumus-rumus matematika.
* *emoji* : jenis font yang digunakan untuk menampilkan emoji.
* *fangsong* : jenis font yang menampilkan gaya penulisan Chinese.

Dalam memilih jenis font terdapat istilah yang dinamakan **web safe font.** Web safe font adalah jenis font yang umumnya sudah pasti tersedia di sebagian besar komputer. Sehingga dapat dipastikan bahwa website akan terlihat sebagaimana mestinya pada browser. Berikut merupakan beberapa contoh font yang masuk ke kategori ini.

* Arial (sans-serif)
* Verdana (sans-serif)
* Helvetica (sans-serif)
* Tahoma (sans-serif)
* Trebuchet MS (sans-serif)
* Times New Roman (serif)
* Georgia (serif)
* Garamond (serif)
* Courier New (monospace)
* Brush Script MT (cursive)

**font-size**

Mengubah nilai *font*pada sebuah dokumen adalah hal yang sangat wajar terjadi, begitu pula pada website. Untuk menetapkan ukuran font, kita perlu menerapkan properti *font-size*. Kita bisa menetapkan nilai dari properti ini dengan menuliskan langsung nilai dan satuannya Pastikan bahwa saat menuliskan nilai dan satuannya, tidak ada jarak (spasi)

Satuan dalam menetapkan ukuran font terdapat dua jenis. Yang pertama *relative*, yakni satuan yang nilainya tergantung pada sesuatu hal, contohnya ukuran dari *viewport*, induk elemen ataupun ukuran teks standar. Dan yang kedua adalah *absolute*, yakni satuan yang nilainya telah ditentukan atau digunakan dalam dunia nyata.

Berikut merupakan nilai satuan yang dapat kita manfaatkan dalam menetapkan ukuran font beserta fungsinya:

**Relative unit**

|  |  |  |
| --- | --- | --- |
| **Satuan** | **Relative to** | **Fungsi** |
| em | Font size | Satuan relatif terhadap ukuran font yang sedang digunakan pada elemen (contohnya, 2em berarti 2 kali lebih besar dari ukuran font seharusnya). |
| ex | Font height | Satuan relatif terhadap tinggi font saat ini, satuan ini sangat jarang sekali digunakan |
| rem | Font size | Mirip seperti em, tetapi rem merupakan satuan relatif terhadap ukuran font dari root element. |
| ch | Font width | Satuan relatif terhadap lebar dari karakter “0” nol. |
| vw | Viewport width | Satuan relatif terhadap 1% lebar viewport. Contoh 1vw = 1% dari lebar viewport. Satuan ini tidak didukung pada browser IE8 ke bawah. |
| vh | Viewport height | Satuan relatif terhadap 1% tinggi viewport. Contoh 1vh = 1% dari tinggi viewport. Satuan ini tidak didukung pada browser IE8 ke bawah. |

**Absolute unit**

|  |  |
| --- | --- |
| **Satuan** | **Fungsi** |
| px | Menetapkan nilai font berdasarkan ukuran pixel |
| pt | Menetapkan nilai font berdasarkan points (1/72 inch di CSS2.1) |
| pc | Menetapkan nilai font berdasarkan picas (1 pica = 12 point) |
| mm | Menetapkan nilai font berdasarkan millimeters |
| cm | Menetapkan nilai font berdasarkan centimeters |
| in | Menetapkan nilai font berdasarkan inches |

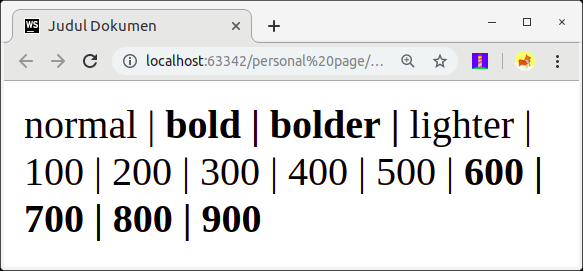
Selain dengan menetapkan nilai dan satuannya secara langsung, untuk mengatur ukuran font kita juga bisa gunakan nilai persentase.

Dan yang terakhir kita juga bisa menentukan ukuran *font*dengan menuliskan kata kunci secara spesifik yang tersedia pada CSS. Kata kunci tersebut adalah: *xx-small, x-small, small, medium, large, x-large*, dan *xx-large*.

Kata kunci tersebut tidak ada kaitannya dengan pengukuran tertentu (bukan ukuran yang *absolute*) tetapi nilainya diubah secara konsisten satu sama lain.

**font-weight**

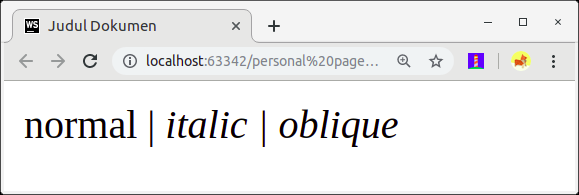
Setelah mengenal *font families* dan *font size*, selanjutnya ada juga *font-weight* yang digunakan untuk mengatur ketebalan dari *font* yang ditampilkan. Nilai dari properti ini dapat ditentukan dengan menggunakan *numeric values* (100 sampai 900) atau dengan menggunakan *descriptive terms* (*normal, bold, bolder*, dan *lighter*).



Properti *font-weight* dapat diaplikasikan ke seluruh elemen yang ada di HTML dan nilainya dapat diturunkan pada elemen turunannya.

**font-style**

Properti selanjutnya adalah *font-style*. Properti ini digunakan untuk menentukan postur dari teks yang ditampilkan, apakah bentuknya vertikal (normal) atau miring (*italic* dan *oblique*).

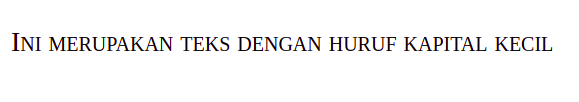


*Italic* dan *oblique* keduanya menampilkan teks yang miring. Perbedaanya adalah *italic* menerapkan tipe miring (*italic font version*) dari suatu font sedangkan *oblique* adalah font normal yang hanya dibuat miring.

Properti *font-style* dapat diaplikasikan ke seluruh elemen yang ada di HTML dan nilainya dapat diturunkan pada elemen turunannya.

**font-variant**

Kita yang terbiasa dengan aplikasi *document editor* (*rich text*) seperti Microsoft Word, tentu tahu atau sudah mencoba fitur *small caps*. Fitur ini dapat membuat teks menjadi kapital tetapi dituliskan secara kecil dan merapat, seperti ini:



Untuk membuat tulisan seperti gambar di atas, kita bisa memanfaatkan properti font-variant dengan nilai small-caps pada propertinya.

Properti *font-variant* dapat diaplikasikan ke seluruh elemen yang ada di HTML dan nilainya dapat diturunkan pada elemen turunannya.

**Shorthand**

Menspesifikasikan masing-masing nilai properti font akan menghasilkan banyak sekali kode repetitif. Dengan begitu CSS memberikan suatu “jalan pintas” untuk menuliskan properti-properti tersebut ke dalam satu properti yaitu *font*.

Dengan menggunakan properti *font* kita dapat menuliskan beberapa properti hanya dalam satu properti pada satu rule.



Nilai dari properti *font* merupakan nilai dari seluruh properti dari font yang sudah kita bahas. Tiap nilai dipisahkan menggunakan spasi. Pada properti ini urutan nilai merupakan hal yang penting, sehingga jangan sampai salah urutan dalam menuliskannya. Kesalahan penulisan atau urutan menyebabkan seluruh struktur rule menjadi tidak valid.

Walaupun begitu kita bisa tidak menuliskan seluruh nilai properti yang ada, tetapi nilai dari properti *font-size* dan *font-family* wajib ada ketika menggunakan properti ini. Berikut contohnya penulisan minimal ketika kita menggunakan properti font:

1. p {
2. font: 1em sans-serif;
3. }

## Text Styling

Pembahasan sebelumnya kita terfokus pada formating bentuk karakter yang ditampilkan dengan menggunakan beberapa properti font yang ada. Sekarang kita akan mempelajari bagaimana seorang developer bisa memberikan formatting pada keseluruhan teks yang ada pada baris paragraf, seperti menetapkan indent, jarak antar baris, kata dan huruf, dan sebagainya. Maka dari itu, mari kita bahas satu persatu.

### Line Height

Properti line-height digunakan untuk mengatur jarak minimal dari garis dasar ke garis dasar dalam menampilkannya teks pada halaman. Jika kita terbiasa dengan software document editor (rich text) seperti Microsoft Word, properti ini mirip dengan fungsi line and paragraph spacing.

Pada penjelasan diatas disebutkan “minimal”, karena jika terdapat sebuah karakter yang tinggi atau besar dalam sebuah baris, maka tinggi dari baris pun akan menyesuaikan agar jarak tetap mengakomodirnya.

Berikut merupakan contoh tiga cara berbeda dalam menerapkan tinggi baris dua kali lebih tinggi dari ukuran font:

1. p {
2. line-height: 2;
3. }
4. p {
5. line-height: 200%;
6. }
7. p {
8. line-height: 2em;
9. }

Cara pertama merupakan cara yang paling mudah digunakan, karena kita dapat menentukan nilai hanya dengan satu angka, di mana angka tersebut nantinya dikalikan dengan nilai font-size sebelum diterapkan pada nilai properti line-height. Contohnya, ukuran font standar pada paragraf adalah 16 pixel. Kita definisikan properti line-height dengan nilai 2, maka nilai properti line-height seharusnya adalah **16 pixel \* 2 = 32 pixel (dua kali lebih besar dari ukuran font).**

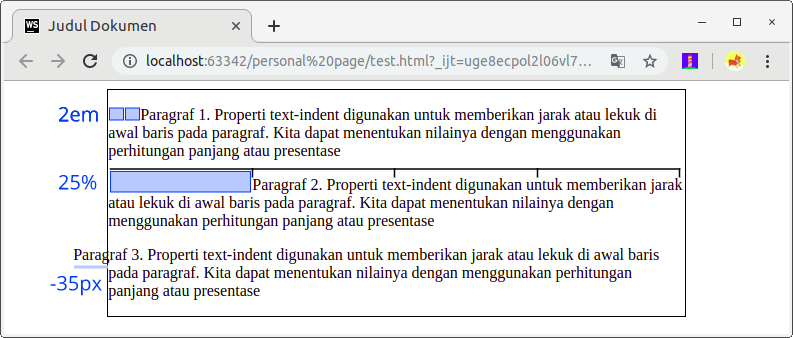
Properti line-height dapat diaplikasikan ke seluruh elemen yang ada pada HTML dan nilainya dapat diturunkan pada elemen turunannya.

### Text Indent

Dalam membuat sebuah dokumen tidak jarang kita membutuhkan jarak/lekuk di awal paragraf. Hal tersebut dapat dilakukan pada website dengan menerapkan properti text-indent.  Kita dapat menentukan nilai properti ini melalui perhitungan panjang dalam px, em, dan in atau bisa menggunakan nilai persentase (%). Nilai persentase dihitung berdasarkan lebar dari induk elemen. Berikut merupakan contoh penggunaannya:

1. p#first {
2. text-indent: 2em;
3. }
5. p#second {
6. text-indent: 25%;
7. }
9. p#third {
10. text-indent: -35px;
11. }

Jika elemen menerapkan rule tersebut, akan tampak seperti ini:



Pada contoh ke-tiga kita bisa melihat bahwa pada nilai properti ini dapat diberikan nilai negatif. Jika kita menggunakannya, maka baris pertama pada paragraf akan keluar dari batas elemen yang menampungnya (biasa disebut hanging indent).

Perlu diingat kembali, properti ini hanya berpengaruh pada awal baris paragraf. Jika kita ingin menetapkannya untuk seluruh baris kita dapat gunakan margin atau padding, keduanya akan dibahas pada materi box model.

### Text Alignment

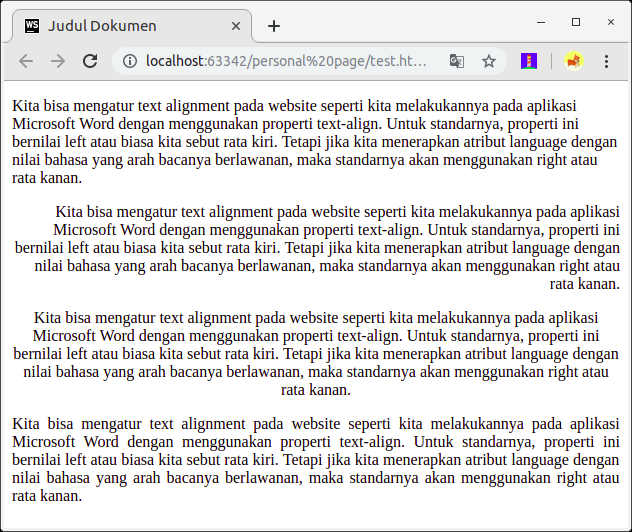
Kita bisa mengatur text alignment (perataan teks) pada website seperti kita melakukannya pada aplikasi Microsoft Word dengan menggunakan properti text-align. Untuk asalnya, properti ini bernilai left atau biasa kita sebut rata kiri. Namun jika kita ingin mengubah perataan teks ini berada di posisi lainnya, maka kita dapat mengubah value-nya menjadi value yang sesuai dengan keinginan kita.

Berikut ini nilai yang dapat digunakan pada properti text-align:

|  |  |
| --- | --- |
| **Nilai Properti** | **Fungsi** |
| text-align: **left** | Membuat perataan teks pada ujung kiri |
| text-align: **right** | Membuat perataan teks pada ujung kanan |
| text-align: **center** | Membuat perataan teks secara menengah |
| text-align: **justify** | Membuat perataan teks yang setara pada ujung kiri dan kanannya |

Bagaimana? Pasti kita sudah familiar dengan nilai-nilai tersebut? Berikut contoh penggunaan dari properti text-align:

1. p#first {
2. text-align: left;
3. }
5. p#second {
6. text-align: right;
7. }
9. p#third {
10. text-align: center;
11. }
13. p#fourth {
14. text-align: justify;
15. }



### Text Decoration

Properti ini paling populer digunakan ketika kita ingin membuat garis bawah atau underline pada teks. Tapi tak hanya itu, ada beberapa nilai lain yang dapat kita gunakan untuk properti ini. Detailnya sebagai berikut:

|  |  |
| --- | --- |
| **Nilai properti** | **Fungsi** |
| text-decoration: **underline** | Memberikan garis bawah (underline) pada teks |
| text-decoration: **overline** | Memberikan garis atas (overline) pada teks |
| text-decoration: **line-through** | Memberikan efek tulisan dicoret (strikethrough) |
| text-decoration: **none** | Menghilangkan dekorasi teks yang ada pada elemen |

Dengan menggunakan text-decoration, kita dapat menghilangkan efek underline pada link yang ditampilkan dengan memberikan nilai text-decoration: none pada elemen <a>.

### Text Transform

Pasti kita pernah mencoba fitur pada document editor yang dapat mengubah kapitalisasi pada teks, bukan? Fitur ini sangat membantu di kala kita ingin mengubah kapitalisasi tanpa harus menuliskan kembali teksnya. Pada CSS juga terdapat fitur serupa, yaitu dengan menggunakan properti text-transform. Ketika kita menerapkan properti text-transform pada elemen teks, maka kapitalisasi akan berubah ketika halaman di-render tanpa harus mengubahnya pada dokumen HTML.

Properti ini dapat berisikan nilai sebagai berikut:

|  |  |
| --- | --- |
| **Nilai Properti** | **Fungsi** |
| text-transform: **none** | Teks yang ditampilkan sama seperti yang dituliskan |
| text-transform: **capitalize** | Membuat huruf pertama besar pada tiap katanya |
| text-transform: **lowercase** | Membuat seluruh teks menggunakan huruf kecil |
| Text-transform: **uppercase** | Membuat seluruh teks menggunakan huruf besar |

Seperti ini contoh tampilan ketika menerapkan properti dan nilai di atas

### Word and Letter Spacing

Properti selanjutnya yang bisa kita gunakan untuk formating teks adalah letter-spacing dan word-spacing. Seperti namanya, properti ini digunakan untuk mengatur spasi atau jarak pada teks. Properti letter-spacing digunakan untuk mengatur jarak antar huruf, sedangkan word-spacing digunakan untuk mengatur jarak antar kata. Berikut contohnya:

1. p#letter {
2. letter-spacing: 4px;
3. }
4. p#word {
5. word-spacing: 1.5em;
6. }

Berikut contoh tampilan jika menerapkan rule di atas:

Kedua elemen ini dapat diaplikasikan ke seluruh elemen yang ada pada HTML dan nilainya dapat diturunkan pada elemen turunannya.

### Text Shadow

Memberikan bayangan pada teks telah menjadi hal yang umum digunakan meskipun tidak memiliki dukungan di semua browser. Pada CSS untuk membuat bayangan pada teks (atau biasa disebut drop shadow) kita dapat gunakan properti text-shadow.

Nilai dari properti ini cukup rumit karena membutuhkan tiga buah nilai dan satu buah nilai warna sehingga membutuhkan empat nilai dalam satu properti untuk menentukan bayangannya.

* Nilai pertama : menunjukkan seberapa jauh ke kiri atau kanan (horizontal) bayangan harus ditampakkan.
* Nilai kedua : menunjukkan jarak ke atas atau ke bawah (vertical) bayangan harus ditampakkan.
* Nilai Ketiga (opsional) : menentukan tingkat keburaman yang harus diterapkan pada bayangan.
* Nilai Keempat :  menentukan warna yang digunakan pada bayangan.

Berikut ini contoh penggunaan dari properti drop shadow:

1. p#one {
2. text-shadow: 1px 1px 0px #000000;
3. background-color: #eeeeee;
4. color: #666666;
5. }
7. p#two {
8. text-shadow: 1px 1px 3px #666666;
9. background-color: #dddddd;
10. color: #666666;
11. }
13. p#three {
14. text-shadow: 2px 2px 7px #111111;
15. background-color: #cccccc;
16. color: #ffffff;
17. }
19. p#four {
20. text-shadow: -1px -1px #666666;
21. background-color: #cccccc;
22. color: #bbbbbb;
23. }

# Color

Warna memberikan kesan hidup pada sebuah website, Jika kita kembali pada tahun 1993 saat website hanya menampilkan background abu dengan teks hitam, tentu membosankan sekali, bukan?

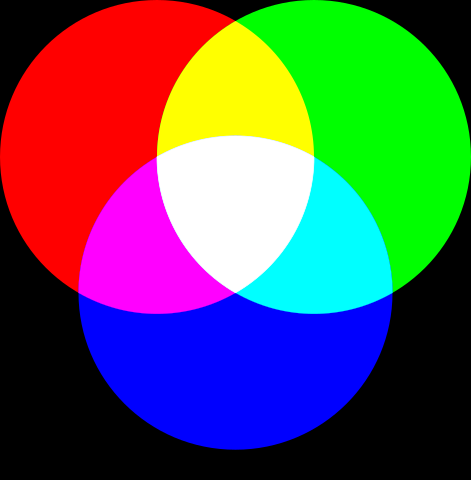
Pada saat itu memang belum tersedia sebuah browser yang dapat mengontrol warna. Sampai akhirnya Netscape Navigator hadir dan memungkinkan hal tersebut, walaupun belum sempurna layaknya saat ini. Syukurlah pada saat ini kita bisa menggunakan properti - properti yang ada pada CSS untuk mengatur warna pada teks dan background. Terlebih, seluruh browser saat ini sudah mendukung styling menggunakan CSS sehingga kita tidak perlu lagi memikirkan kompatibilitasnya.

Pembahasan kali ini kita akan fokus mengenai warna pada CSS. Anda akan dikenalkan pada properti yang digunakan untuk mengatur warna teks dan background. Tapi sebelum itu, mari kita pelajari dahulu bagaimana cara menetapkan sebuah warna pada CSS.

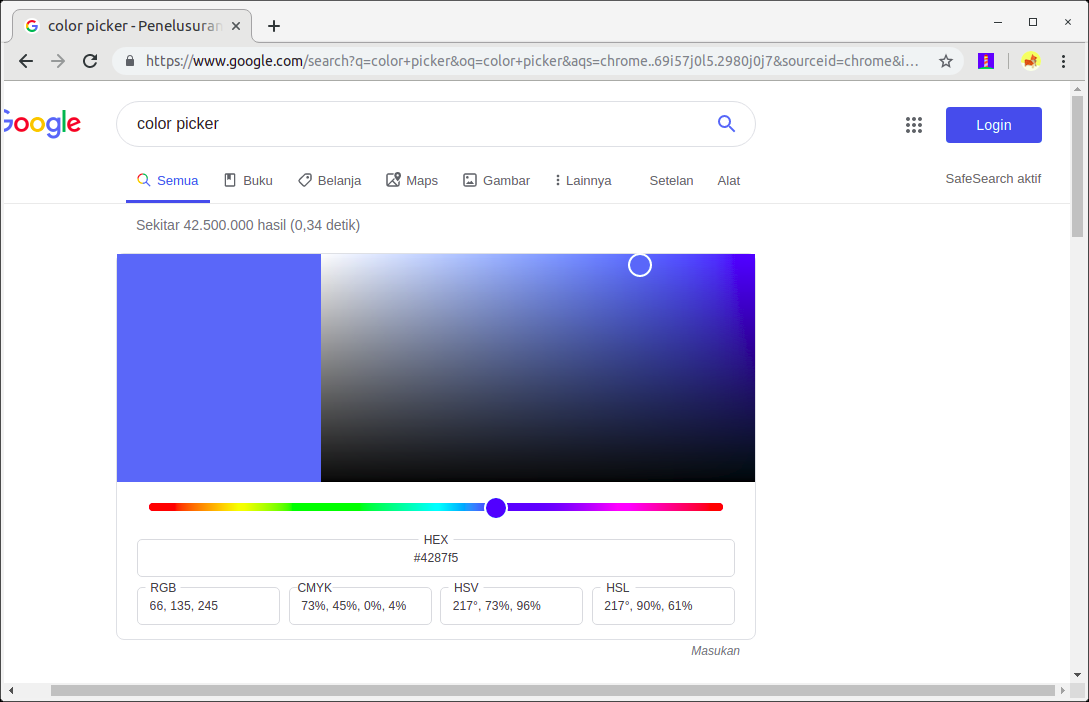
## Menetapkan Nilai Warna

Setiap warna pada layar komputer kita terdiri dari campuran warna merah, hijau, dan biru. Layar komputer dibuat dari ribuan kotak kecil yang biasa disebut pixel. Ketika layar monitor tidak menyala maka pixel pun tidak akan menyala. Ketika layar monitor menyala, tiap pixel dapat menghasilkan warna yang berbeda sehingga dapat menampilkan gambar.

Di dalam pixel terdapat tiga buah lampu kecil berwarna merah, hijau, dan biru. Ribuan warna dapat dihasilkan dari kombinasi tiga warna tersebut.



ketika ingin menetapkan warna, kita tidak perlu menghafal kombinasi dari ketiga warna tersebut. Banyak pemilih warna yang dapat membantu kita dalam menetapkan warna yang diinginkan. Kita bisa melihat nilai RGB melalui color picker. Lantas bagaimana jika kita tidak memiliki aplikasi design seperti yang telah disebutkan tadi? Tenang, sekarang color picker banyak tersedia online, bahkan ketika kita menuliskan “color picker” pada Google Search, tools tersebut tersedia pada hasil pencariannya.



Untuk menetapkan nilai warna pada CSS kita dapat menggunakan Numeric Value dan Predefined Color Name.

### Numeric Value

Cara yang paling banyak digunakan dalam menetapkan warna adalah  dengan menggunakan numeric value. Apa pasal? Dengan menggunakannya kita dapat menentukan warna yang kita inginkan dengan tepat. Terlebih dengan bantuan color picker maka kita tak perlu repot memahami dan mencari nilai kode warna secara manual.

Berikut contoh penulisan nilai warna pada CSS dengan numeric value:

1. /\* Warna hijau menggunakan format RGB \*/
2. color: rgb(78,231,23);
4. /\* Warna hijau Menggunakan format HEX \*/
5. color: #4ee717;
7. /\* Warna hijau menggunakan format HSL (hanya di CSS3)\*/
8. color: hsl(104, 82%, 50%);

Kode tersebut merupakan nilai warna hijau yang dituliskan dalam beberapa format. Dari kode tersebut kita tahu tidak hanya format RGB yang dapat digunakan dalam menetapkan warna. Selain RGB ada juga format Hex dan HSL (Hue, Saturation, Light).

### Predefined Color Name

Menentukan warna bisa juga dilakukan selain dengan menggunakan format angka. Kita bisa menggunakan sebuah kata seperti blue, yellow, red dan lainnya

1. color: black;
2. color: white;
3. color: blue;

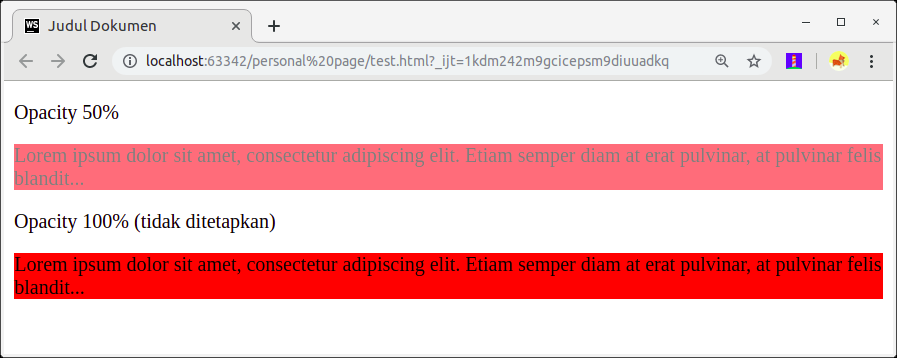
Dengan menggunakan kata tentu kita dapat lebih mudah dan cepat dalam menentukan warna, tetapi mungkin hanya warna dasar yang sudah kita hafal. Faktanya, varian warna ada banyak jumlahnya! Hampir seluruh browser saat ini mendukung 140 nama warna, wow! Kita dapat melihat apa saja warna yang tersedia pada tautan berikut: <https://www.w3schools.com/colors/colors_names.asp>.

### Opacity

CSS3 mengenalkan properti opacity yang dapat menspesifikasikan nilai transparan pada elemen HTML. Nilai dari properti opacity merupakan angka desimal antara 0.0 hingga 1.0, jika kita menetapkan nilainya 0.5 itu berarti 50% transparan.

1. div#opacity {
2. background-color: red;
3. opacity: 0.5;
4. }

Jika elemen div menerapkan rule tersebut, maka akan tampak seperti ini:



Selain pada elemen, menetapkan opacity juga dapat kita lakukan pada warna. Untuk melakukannya bisa menggunakan format **rgba,** dan **hsla**, di mana format ini merupakan format rgb ditambahkan nilai alpha. Selain itu, dapat juga dilakukan menggunakan format **hex** dengan memberikan dua digit dibelakang kode *hexadecimal.*

1. p {
2. color: rgba(78,231,23, 0.5);
3. color: hsla(104, 82%, 50%, 0.5);
4. color: #4ee717ee; /\*ee merupakan digit tambahan untuk menerapkan opacity\*/
5. }