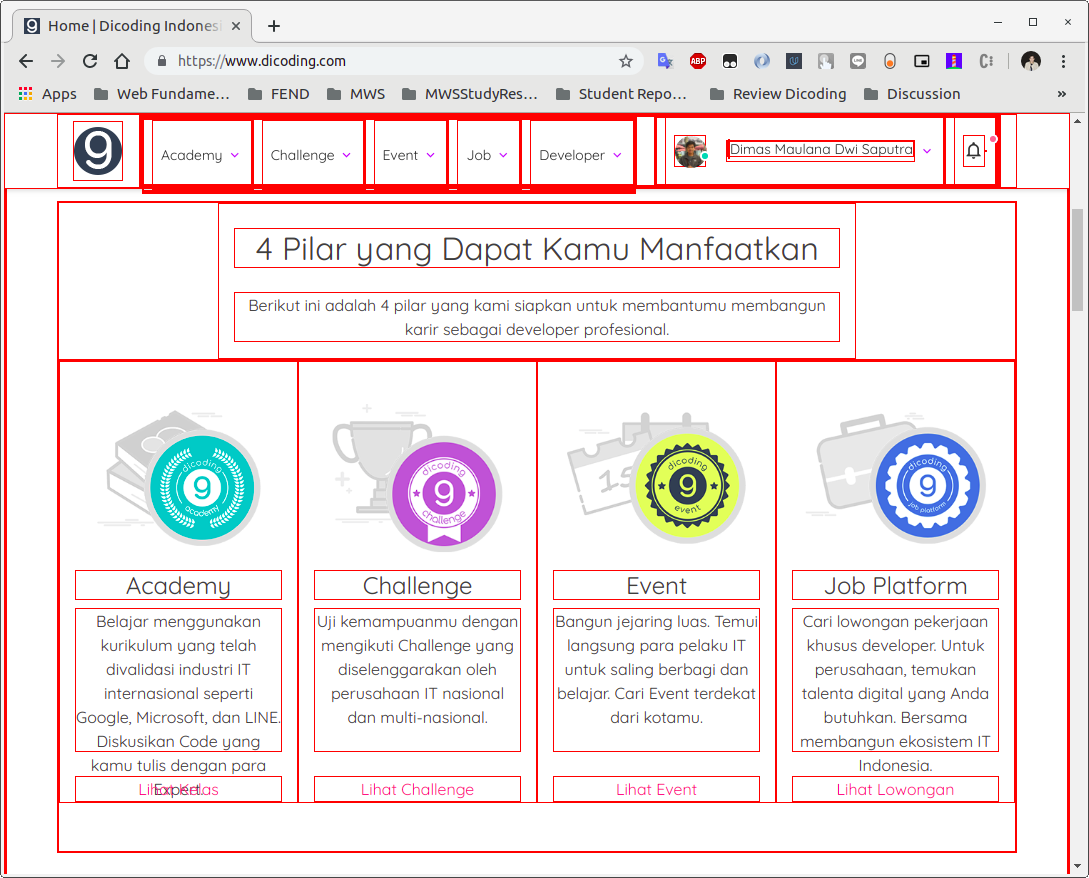
# Box Model

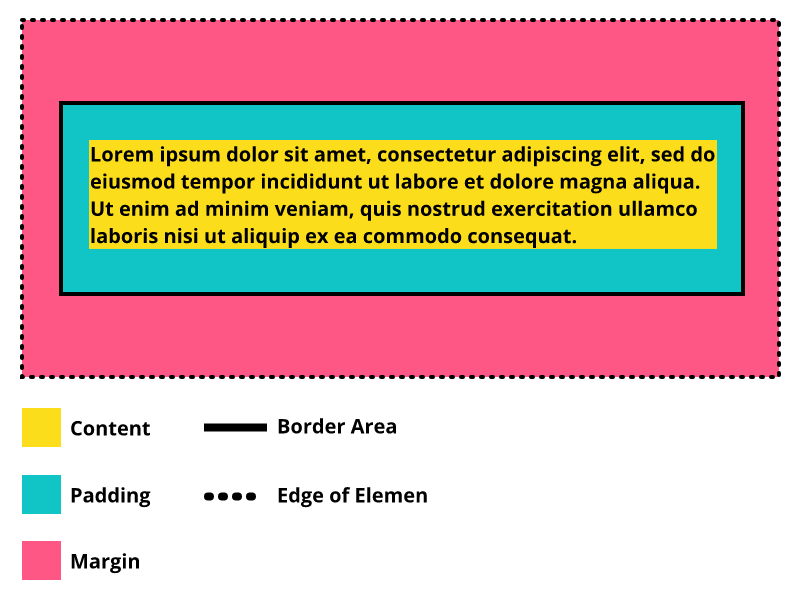
Pada pembahasan background-color disinggung bahwa setiap elemen ditampilkan seperti sebuah boxes atau kotak. Box model merupakan salah satu konsep dari CSS. Setiap elemen yang dibuat pada HTML akan menciptakan sebuah kotak untuk menampung kontennya.  Layaknya bentuk kotak pada umumnya, pasti terdapat nilai seperti width (lebar), height (tinggi), padding (tebal lapisan), border (garis tepi), dan juga jarak. Pada CSS nilai - nilai tersebut dapat kita atur.

Kita mungkin sudah sedikit paham bagaimana konsep box model ini bekerja dengan mengikuti latihan sebelumnya. Di sini kita akan membahas lebih lanjut mengenai properti apa saja yang dapat memanipulasi boxes tersebut. Tapi sebelum itu, kita pahami lebih tentang konsep box atau kotak pada elemen HTML agar kita dapat gambaran lebih tentang hal tersebut.

## Box Element



Seperti yang kita lihat pada gambar di atas, setiap elemen pada HTML baik itu block-level atau inline elemen, akan menghasilkan sebuah kotak elemen. Perhatikan ilustrasi berikut agar lebih jelasnya:



Content : Merupakan sebuah inti dari kotak elemen yang merupakan konten dari elemen itu sendiri

* Padding : Area yang menjadi jarak antara border elemen dengan konten yang ditampilkan. Ketika kita menerapkan background-color maka warna tersebut diterapkan pada area padding. Penerapan ukuran padding pada elemen merupakan opsional.
* Border : Merupakan garis terluar yang mengelilingi konten dan padding. Penerapan border pada elemen merupakan opsional.
* Margin : Berbeda dengan padding, margin merupakan area jarak diluar dari border. Pada ilustrasi diatas, margin ditunjukkan dengan warna merah muda namun pada aslinya margin selalu transparan dan penerapan margin pada elemen merupakan opsional.
* Edge of element : merupakan batas dari suatu elemen. Di dalamnya terdapat margin, border, padding dan konten. Pada ilustrasi di atas, batas elemen ditunjukkan dengan garis putus tetapi pada aslinya batas elemen selalu tidak terlihat.

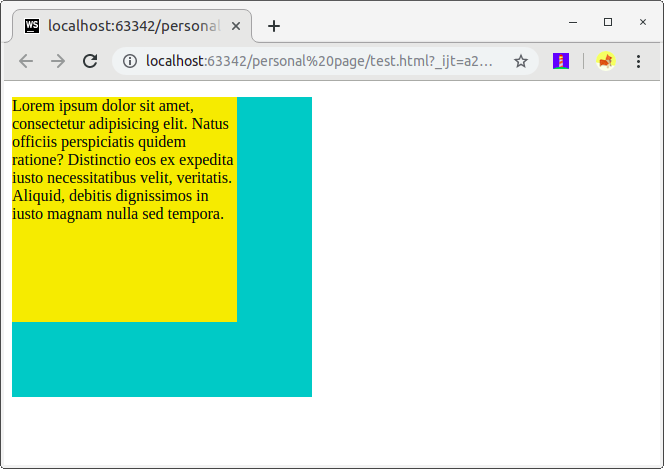
## Box Dimensions

Secara standar sebuah box yang dihasilkan tiap elemen selalu cukup untuk menampung konten. Tetapi kita dapat mengatur nilai dimensi dari box tersebut dengan properti width dan height.

Cara yang paling banyak digunakan dalam menentukan dimensi kotak adalah dengan menggunakan pixel, persentase, atau ems. Secara tradisional, pixel merupakan cara yang paling populer karena kita dapat merancang dan mengontrol ukuran secara akurat.

Berbeda ketika kita menggunakan persentase, ukuran kotak akan relative atau menyesuaikan dari ukuran lain, seperti ukuran jendela browser atau ukuran induk yang menaunginya. Sedangkan ketika menggunakan ems, nilai dimensi kotak akan menyesuaikan berdasarkan ukuran teks yang ditampilkan pada konten elemen tersebut.

Pada saat ini banyak developer mulai merancang menggunakan persentase dan ems untuk menetapkan ukuran box agar dapat menyesuaikan dengan berbagai macam ukuran layar.



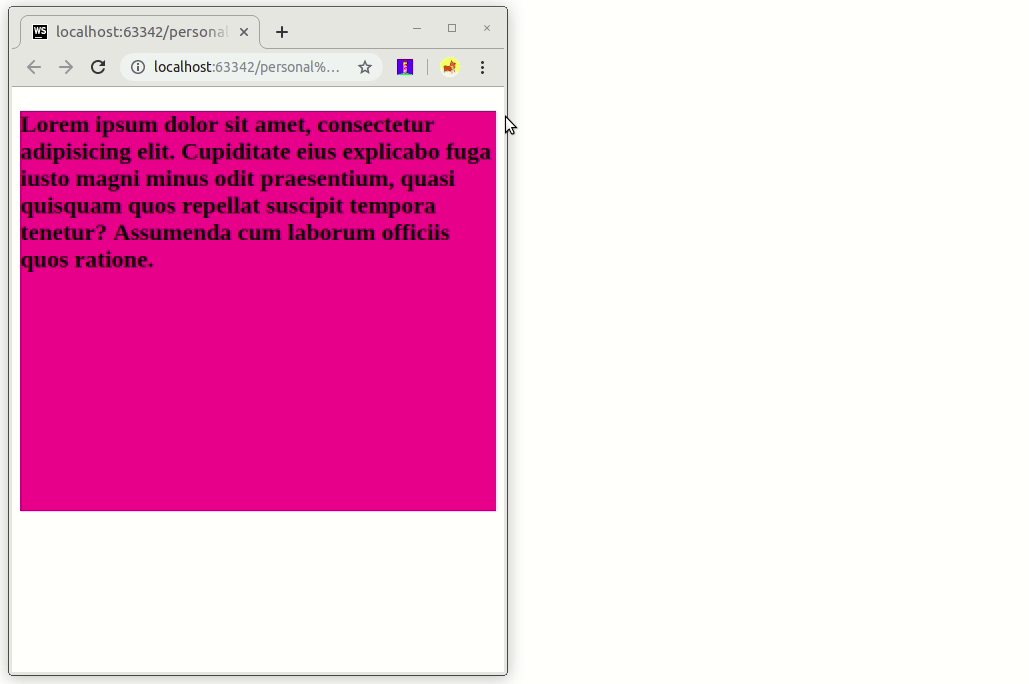
Pada contoh di atas kita dapat melihat elemen <div> memiliki dimensi elemen dengan lebar 300px dan tinggi 300px. Di dalamnya terdapat elemen <p> yang memiliki ukuran elemen 75% dari lebar dan tinggi elemen induknya. Dengan begitu berarti elemen <p> memiliki ukuran 225px untuk panjang dan lebarnya.

### Limiting Dimension

Beberapa website yang ada sekarang menampilkan layout yang dapat melebar dan menyempit mengikuti ukuran layar pengguna. Pada prinsip tampilan tersebut mungkin kita memerlukan sebuah limitasi ukuran yang harus ditampilkan agar konten selalu dapat ditampilkan secara proporsional. Untuk melakukannya kita manfaatkan properti min-width dan max-width.

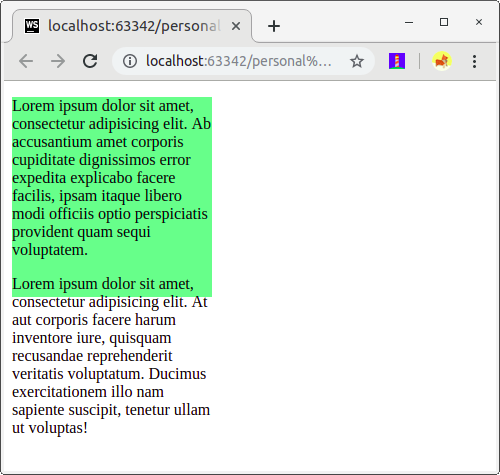
* min-width : merupakan properti yang digunakan untuk menetapkan nilai lebar minimal yang harus dimiliki elemen.
* max-width : merupakan properti yang digunakan untuk menetapkan nilai lebar maksimal yang harus dimiliki elemen.

Keduanya merupakan properti yang sangat membantu untuk memastikan konten halaman dapat terbaca oleh pengguna (terutama ketika pengguna menggunakan ponsel). Misalnya, kita dapat menggunakan properti max-width untuk memastikan bahwa baris teks yang muncul tidak terlalu lebar.



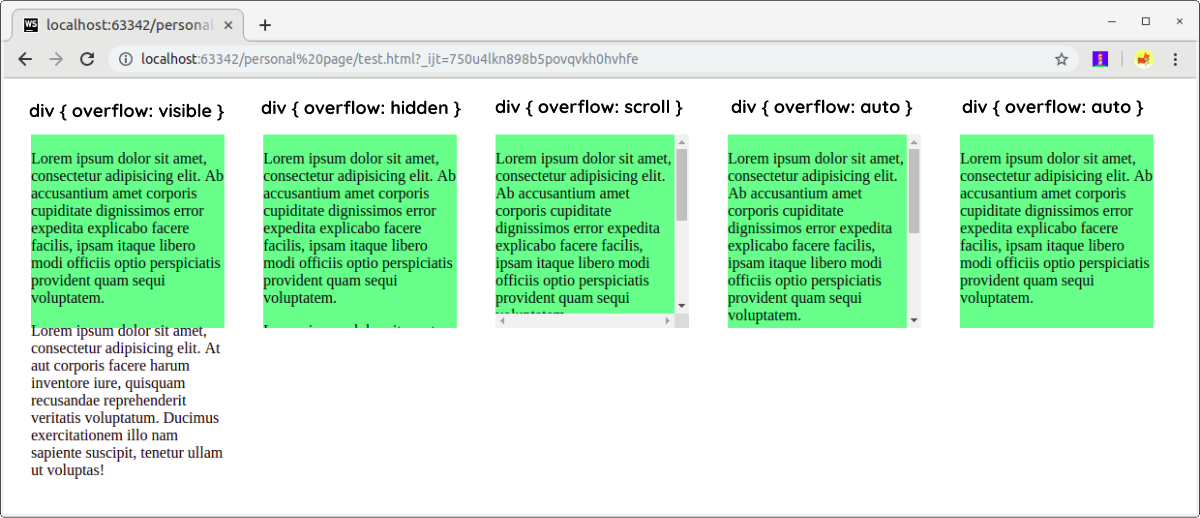
Dengan cara yang sama, mungkin kita juga perlu membatasi ukuran panjang. Kita bisa gunakan min-height dan max-height.

### Overflowing Content

Dimensi box yang dihasilkan elemen selalu cukup untuk menampung konten tetapi hal ini tidak berlaku jika kita tetapkan secara manual panjang dan lebarnya. Tak jarang terjadi overflow ketika kita menerapkan ukuran pada elemen tetapi konten di dalamnya begitu banyak. Contohnya seperti berikut:

Untuk menangani kasus seperti ini kita bisa gunakan properti overflow. Properti ini dapat bernilai:

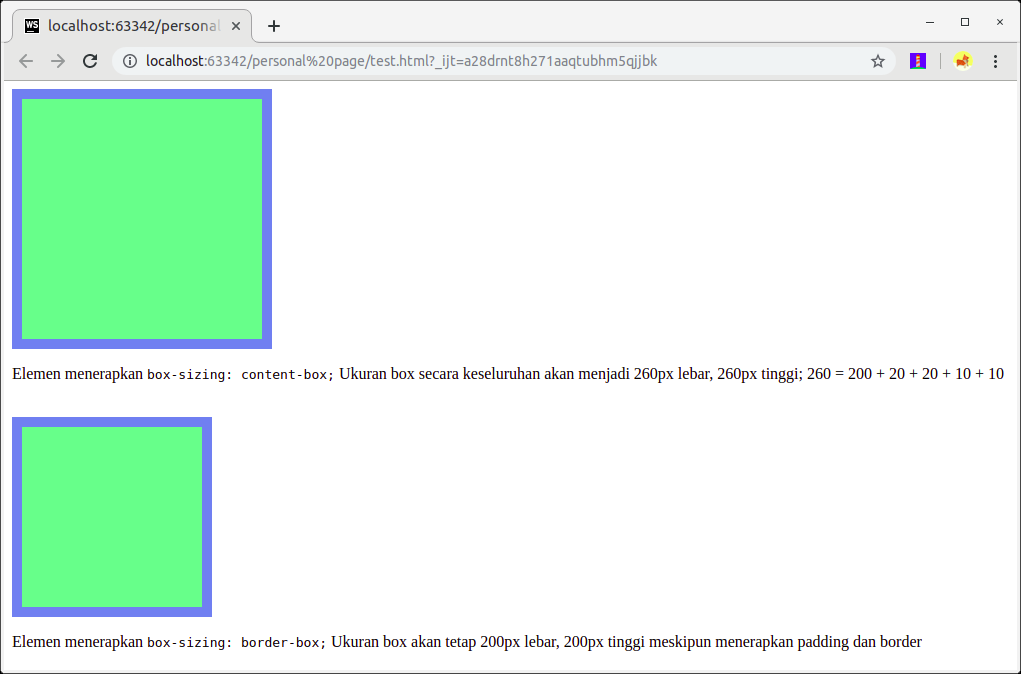
* visible : Merupakan nilai default pada properti ini. Konten yang tidak tertampung (overflow) akan tetap ditampilkan seperti pada standarnya.
* hidden : Jika terjadi overflow, maka konten yang tidak tertampung akan disembunyikan.
* scroll : Memunculkan scroll bar pada pinggir elemen, sehingga konten yang tidak tertampung akan ditampilkan dengan scroll bar. Jika menggunakan nilai ini, scroll bar akan tetap muncul walaupun konten tidak terjadi overflow.
* auto : Sama seperti scroll, bedanya jika tidak terjadi overflow, maka nilai visible yang akan diterapkan.



### Box-sizing

Pada CSS2 ukuran lebar dan panjang elemen mengacu pada konten elemen (content-box). Itu berarti ukuran elemen seluruhnya merupakan nilai panjang dan lebar yang kita spesifikasikan ditambah dengan nilai padding dan border yang diterapkan pada elemen. Hal tersebut membuat sebagian developer menjadi sulit menetapkan ukuran dimensi.

Setelah CSS3 kita dapat memilih tipe pengukuran lain dalam menentukan dimensi elemen. Dengan menggunakan properti box-sizing kita dapat menentukannya berdasarkan border box, di mana ukuran elemen sudah termasuk content, padding dan border. Dengan metode ini, hasil elemen yang ditampilkan (termasuk padding dan border) akan memiliki dimensi yang sama persis seperti yang kita tentukan.



## Border

Border merupakan sebuah garis yang mengelilingi area konten dan padding (opsional). Kita bisa mengatur tipe, ketebalan, serta warna garis yang ditampilkan sesuai dengan yang kita inginkan. Kita juga bisa mengatur dalam menampilkan sebagian atau keseluruhan garis pada elemen. Mari kita eksplorasi apa saja properties yang dapat mengatur border.

### Border Width

Properti border-width digunakan untuk mengatur ketebalan dari border. Nilai dari properti ini dapat berupa pixel atau menggunakan predefined names value seperti thin, medium, dan thick. Kita tidak bisa menggunakan nilai persentase (%) pada properti ini.

Properti border-width dapat ditentukan dengan menggunakan satu, dua, tiga, atau empat nilai. Berikut penjelasannya:

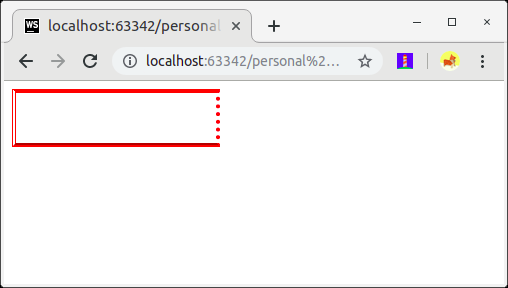
* Ketika satu nilai ditentukan, maka nilai berlaku untuk empat sisi.
* Ketika dua nilai ditentukan, nilai pertama berlaku untuk sisi atas dan bawah, nilai kedua untuk sisi kiri dan kanan.
* Ketika tiga nilai ditentukan, nilai pertama berlaku untuk sisi atas, nilai yang kedua untuk sisi kiri dan kanan, nilai ketiga untuk sisi bawah.
* Ketika empat nilai ditentukan, nilai pertama berlaku untuk sisi atas, nilai yang kedua untuk sisi kanan, nilai yang ketiga untuk sisi bawah, dan nilai yang keempat untuk sisi kiri. Urutan tersebut berdasarkan arah jarum jam (*clockwise*).

### Border Style

Kita bisa menetapkan tipe border dengan menggunakan properti border-style. Berikut nilai - nilai yang dapat digunakan pada properti ini:

|  |  |  |
| --- | --- | --- |
| **Nilai Properti** | **Penjelasan** | **Gambar** |
| solid | Tipe garis padat (tidak terputus - putus) |  |
| dotted | Garis yang dibentuk dari serangkaian titik-titik (jika ketebalan garis 2px, maka titik-titik akan berukuran 2px dan memiliki jarak 2px antar titiknya). |  |
| dashed | Garis yang dibentuk dari serangkaian garis pendek. |  |
| double | Garis yang dibentuk dari dua buah garis padat. |  |
| groove | Tipe garis yang berbentuk seperti frame |  |
| hidden | Digunakan untuk menyembunyikan garis pada elemen. |  |

Kita juga bisa menetapkan tipe garis secara individual pada sisi elemen dengan menggunakan empat properti terpisah. Contohnya seperti ini:

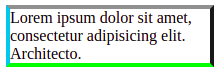


### Border Color

Properti terakhir adalah border-color. Properti ini digunakan untuk menentukan warna pada garis dengan menggunakan nilai RGB, Hex, atau nama warna pada CSS.

Sama seperti properti border yang lain, kita dapat menentukan warna pada individual sisi pada elemen dengan menggunakan properti yang terpisah.

Tetapi kita juga dapat menetapkan nilai keempatnya sekaligus dalam satu properti seperti ini:



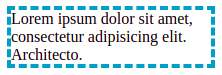
### Shorthand

Untuk menerapkan border pada elemen kita harus mendefinisikan seluruh properti border yang ada. Dimulai dari menetapkan ketebalan (border-width), tipe (border-style), dan warna (border-color). Jika kita lupa menetapkan salah satu properti tersebut, maka garis tidak akan nampak pada elemen.

Dengan begitu tentu untuk menetapkan border pada elemen, kita perlu menuliskan properti yang banyak bukan? Ya memang, tetapi CSS menyediakan jalan pintas (shorthand) untuk membuat border dengan satu properti saja. Properti border memiliki tiga buah nilai yang digunakan untuk menentukan ketebalan, tipe dan warna pada border. Berikut contoh penerapannya:

1. .box {
2. border: 4px dashed #00a2c6;
3. width: 200px;
4. }

Hasilnya seperti ini:



Perlu kita perhatikan bahwa penulisan urutan harus benar. Nilai pertama digunakan untuk ketebalan, yang kedua untuk tipe, dan yang ketiga untuk warna garis.

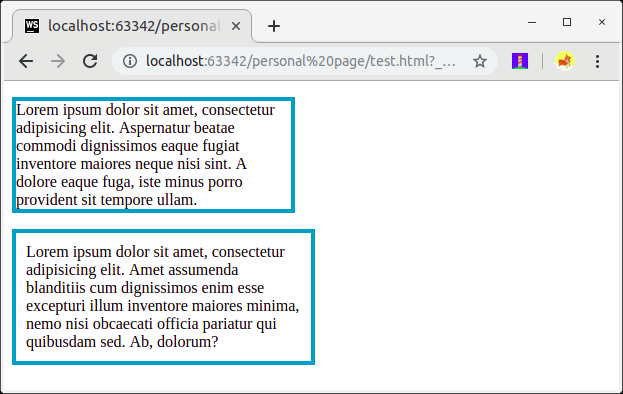


## Padding

Padding merupakan jarak antara area konten dan border. Padding banyak diterapkan pada elemen jika elemen tersebut menerapkan warna latar atau pun border. Padding memberikan sedikit ruang sehingga konten di dalam elemen dapat lebih nyaman untuk ditampilkan. Contohnya:

1. <!doctype html>
2. <html lang="en">
3. <head>
4. <style>
5. p {
6. border: 4px solid #00a2c6;
7. width: 275px;
8. }
10. p.example {
11. padding: 10px;
12. }
13. </style>
14. </head>
15. <body>
16. <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Aspernatur beatae commodi dignissimos eaque fugiat inventore maiores neque nisi sint. A dolore eaque fuga, iste minus porro provident sit tempore ullam.</p>
17. <p class="example">Lorem ipsum dolor sit amet, consectetur adipisicing elit. Amet assumenda blanditiis cum dignissimos enim esse excepturi illum inventore maiores minima, nemo nisi obcaecati officia pariatur qui quibusdam sed. Ab, dolorum?</p>
18. </body>
19. </html>

Jika berkas di atas dijalankan pada browser, maka akan tampak seperti ini:



Seperti yang kita lihat, paragraf kedua jauh lebih nyaman dibaca karena terdapat jarak antara teks dan border dari kotak. Kotak pun menjadi sedikit lebar karena adanya padding.

Pixel merupakan satuan yang sering digunakan dalam menetapkan nilai properti ini (meskipun kita bisa juga menggunakan persentase atau ems). Jika menetapkan menggunakan persentase, maka nilai akan menjadi relatif pada elemen induk atau jendela browser (jika tidak memiliki induk elemen).

Kita dapat menentukan nilai padding yang berbeda untuk masing-masing sisi elemen dengan menggunakan:

*padding-top: 10px;*

*padding-right: 15px;*

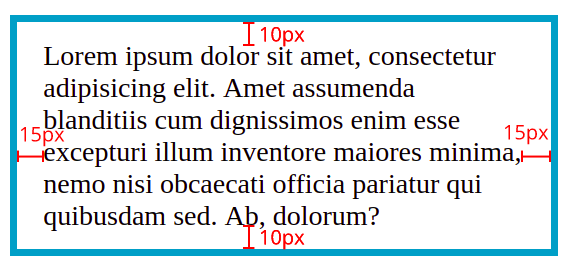
*padding-bottom: 10px;*

*padding-left: 15px;*

Atau dengan menggunakan shorthand seperti berikut:

*padding: 10px 15px 10px 15px;*

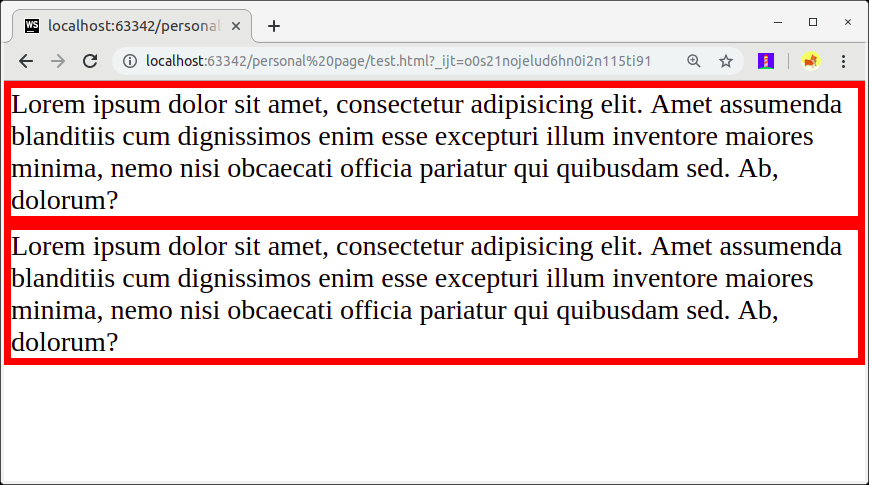
Sehingga padding akan diterapkan seperti gambar di bawah ini.

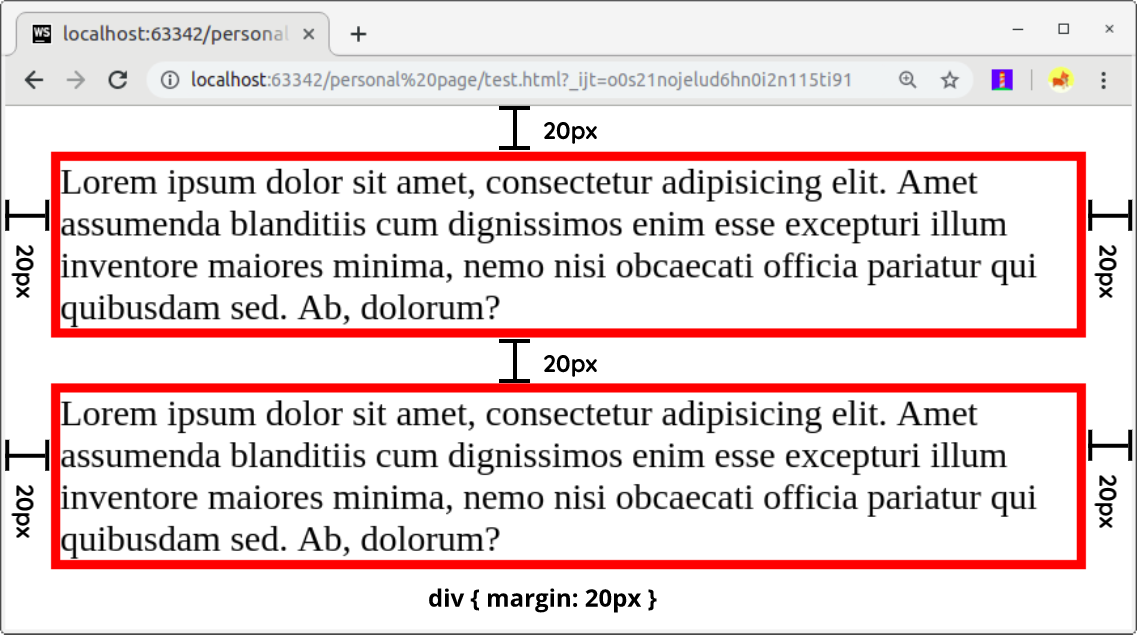


## Margin

Seperti halnya padding, margin merupakan ruang atau jarak pada sebuah elemen. Namun jarak tersebut terletak di luar dari konten dan border elemen. Margin digunakan untuk menjaga elemen agar tidak bertabrakan satu sama lain atau dari tepi jendela browser.

Berikut sebelum menambahkan margin



Setelah menerapkan margin

Margin ini bersifat collapsed, yang artinya dapat menumpuk jika terdapat dua margin yang saling bertumpukan. Anda bisa melihat pada margin bawah dan margin atas pada kedua elemen tersebut. Hal itu menjelaskan mengapa jarak vertikal antar elemen tersebut tidak 40px melainkan 20px. Jika terjadi pertumpukan margin, maka nilai yang paling besar yang akan diterapkan.

Kita dapat menentukan nilai margin yang berbeda untuk masing-masing sisi elemen dengan menggunakan:

margin-top: 10px;margin-right: 15px;margin-bottom: 20px;margin-left: 25px;

Tapi kita juga dapat menggunakan shorthand untuk menetapkan keempat nilai tersebut dalam satu properti

margin: 10px 15px 20px 25px;

Kita juga bisa menggunakan dua nilai saja untuk menentukan nilai margin vertikal dan horizontal.

margin: 10px 15px; /\* top dan bottom bernilai 10px, left dan right bernilai 15px \*/

## Centering Content

Bagaimana meletakkan sebuah kotak tepat pada tengah sebuah halaman atau di dalam elemen induknya? Aturlah margin kanan dan kiri dengan nilai auto. Lebih lanjut, kita juga harus menentukan lebar dari kotak tersebut (menggunakan properti width). Jika tidak, kotak akan mengambil lebar penuh pada halaman atau induk elemen.

Setelah kita menentukan lebar kotak dan mengatur margin kiri dan kanan menjadi auto, maka secara otomatis browser akan memberi jarak yang sama di setiap sisi horizontal kotak, sehingga membuat kotak berada di tengah halaman.

Berikut contohnya:

.center {

margin: 0 auto;

}

.box {

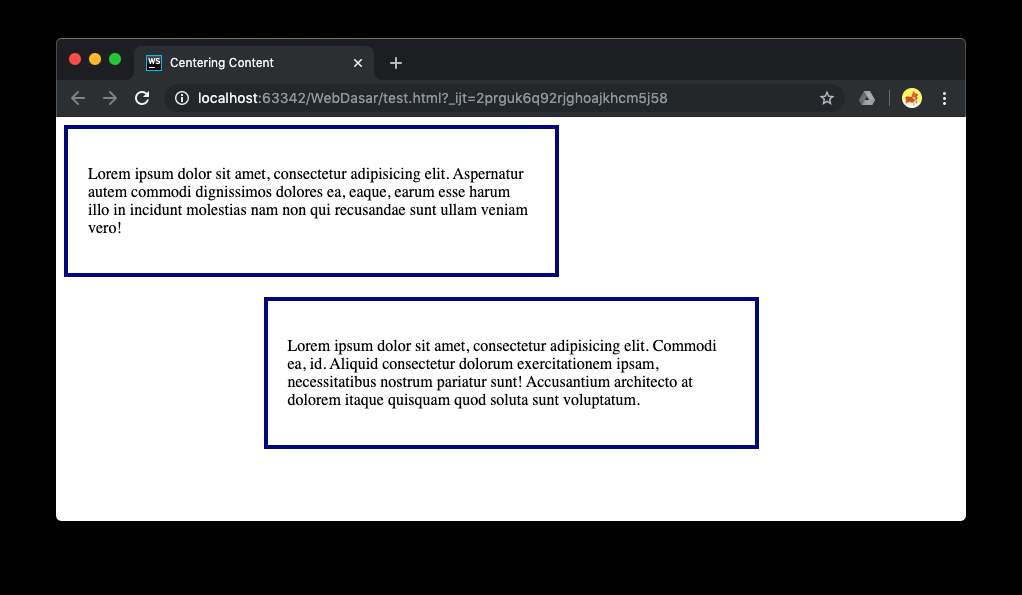
width: 50%;

border: 4px solid darkblue;

padding: 20px;

margin-bottom: 20px;

}



## Display Roles

Kita pernah membahas mengenai inline dan block pada sub-modul Pendalaman HTML. Kita telah paham bahwa standarnya, tiap elemen HTML memiliki dua tipe yaitu block dan inline. Untuk lebih jelasnya berikut sifat - sifat yang ada pada elemen block dan juga inline:

inline element:

* Elemen HTML yang secara default tidak menambahkan baris baru ketika dibuat.
* Nilai lebar dan tinggi elemen inline sebesar konten di dalamnya, dan tidak dapat diubah.
* Margin dan padding hanya mempengaruhi elemen secara horizontal, tidak vertikal.

block element:

* Elemen HTML secara default menambahkan baris baru ketika dibuat.
* Jika tidak diatur lebarnya, lebar dari elemen block akan memenuhi lebar dari browser atau elemen yang menaunginya.
* Kita dapat mengatur dimensi dari elemen block.
* Di dalam elemen block, kita dapat menyimpan tag elemen HTML lainnya.

Dengan menggunakan properti display, kita dapat mengubah sebuah elemen inline menjadi block, begitupun sebaliknya. Dalam hal ini, properti ini juga dapat digunakan untuk menyembunyikan elemen yang ditampilkan.

Nilai dari properti ini dapat berupa:

* inline : digunakan untuk mengubah elemen block berperilaku seperti elemen inline.
* block : digunakan untuk mengubah elemen inline berperilaku seperti elemen block.
* inline-block : membuat elemen block tidak menambahkan baris baru ketika dibuat, namun tetap mempertahankan sifat lain dari elemen block.
* none : digunakan untuk menyembunyikan elemen dari halaman.

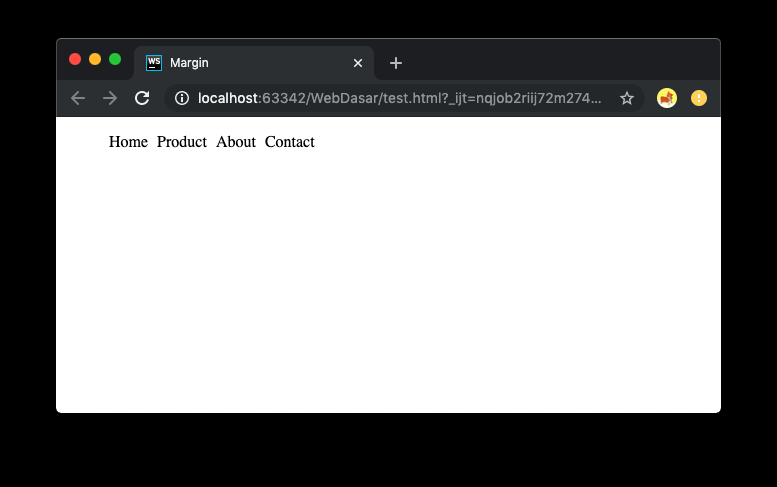
Properti ini banyak sekali digunakan dalam kasus pembuatan navigasi. Biasanya navigasi dibuat menggunakan elemen list yang memiliki sifat block, sehingga item list selalu ditampilkan dalam baris baru. Dengan menggunakan properti display, kita dapat mengubah perilaku elemen list tersebut dengan menerapkan inline, sehingga item list dapat ditampilkan secara horizontal.

li {

display: inline;

margin-left: 5px;

}



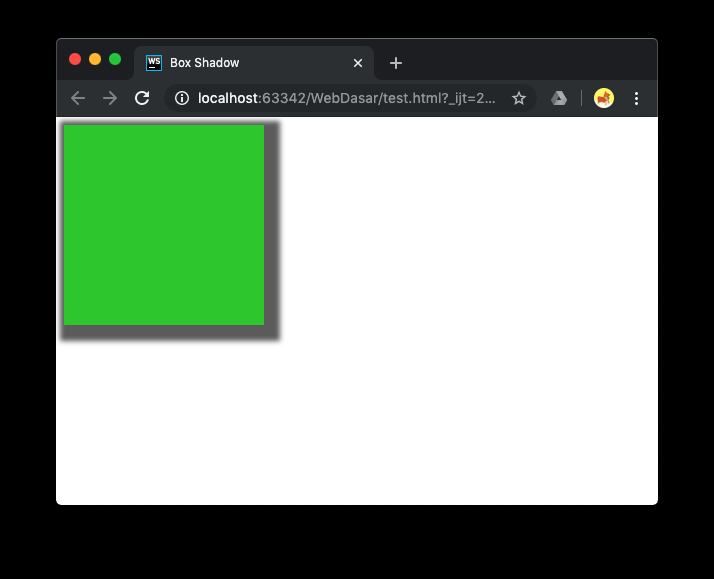
## Box Shadow

Pada materi formatting text kita sudah belajar cara menambahkan drop shadow pada teks. Pada CSS3 kita juga dapat menambahkan drop shadow di sekitar kotak elemen (tidak termasuk margin) dengan menggunakan properti box-shadow.

Nilai dan cara kerja dari properti box-shadow mirip seperti text-shadow, yaitu nilainya menentukan jarak vertikal dan horizontal, tingkat keburaman, dan warna pada bayangan. Pada box shadow, kita juga dapat menentukan tingkat sebaran (spread) bayangan. Semakin besar  nilai, bayangan yang nampak akan semakin luas. Berikut contoh penerapan box shadow pada CSS:

box-shadow: 6px 6px 5px 10px #666666;

Jika rules tersebut diterapkan pada box, maka akan tampak seperti ini:



Berikut penjelasan tiap-tiap nilai dari propertinya:

* 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 (opsional) : menentukan tingkat sebaran (spread) bayangan. Semakin besar nilai yang ditentukan, bayangan yang nampak pun semakin luas.
* Nilai Kelima : menentukan warna yang digunakan pada bayangan.