

CSS (*Cascading Style Sheets*) adalah kode yang kamu gunakan untuk memberikan gaya pada halaman web kamu. *Dasar-dasar CSS* akan menjelaskan apa sajakah yang kamu butuhkan untuk memulai. Kami akan menjawab pertanyaan-pertanyaan seperti: Bagaimana saya dapat membuat teks saya menjadi berwarna hitam atau merah? Bagaimana saya dapat membuat konten saya tampil sedemikian rupa pada layar? Bagaimana saya dapat mendekor halaman web saya dengan latar belakang gambar atau warna-warna?

Apakah CSS itu?

Seperti HTML, CSS bukanlah sebuah bahasa pemrograman. CSS juga bukanlah sebuah bahasa *markup* — ia adalah suatu bahasa *style sheet*. Artinya, dengan CSS kamu dapat mengaplikasikan gaya pada elemen-elemen yang ada dalam dokumen HTML. Sebagai contoh, untuk membuat teks pada seluruh elemen paragraf dalam satu halaman HTML menjadi berwarna merah, kamu akan menulis CSS sebagai berikut:

```
1 | p {  
2 |   color: red;  
3 | }
```

Salin ketiga baris dari kode CSS tersebut ke dalam suatu *file* baru di teks editor kamu, lalu simpan *file* tersebut sebagai `style.css` di direktori `styles` kamu.


Namun kita masih perlu untuk mengaplikasikan CSS tersebut ke dokumen HTML kamu. Jika tidak, CSS tersebut tidak akan berpengaruh pada bagaimana *browser* akan menampilkan dokumen HTML tersebut. (Kalau kamu belum mengikuti proyek kami, bacalah [Dealing with files](#) dan [HTML basics](#) untuk mencari tahu apa saja yang perlu kamu lakukan sebelumnya.)

1. Buka *file* `index.html` kamu dan letakkan baris berikut di suatu tempat di bagian *head* (di antara *tag* `<head>` dan `</head>`):

```
1 | <link href="styles/style.css" rel="stylesheet" type="text/css">
```

2. Simpan `index.html` dan buka halaman tersebut di *browser* kamu. Seharusnya muncul tampilan seperti ini:

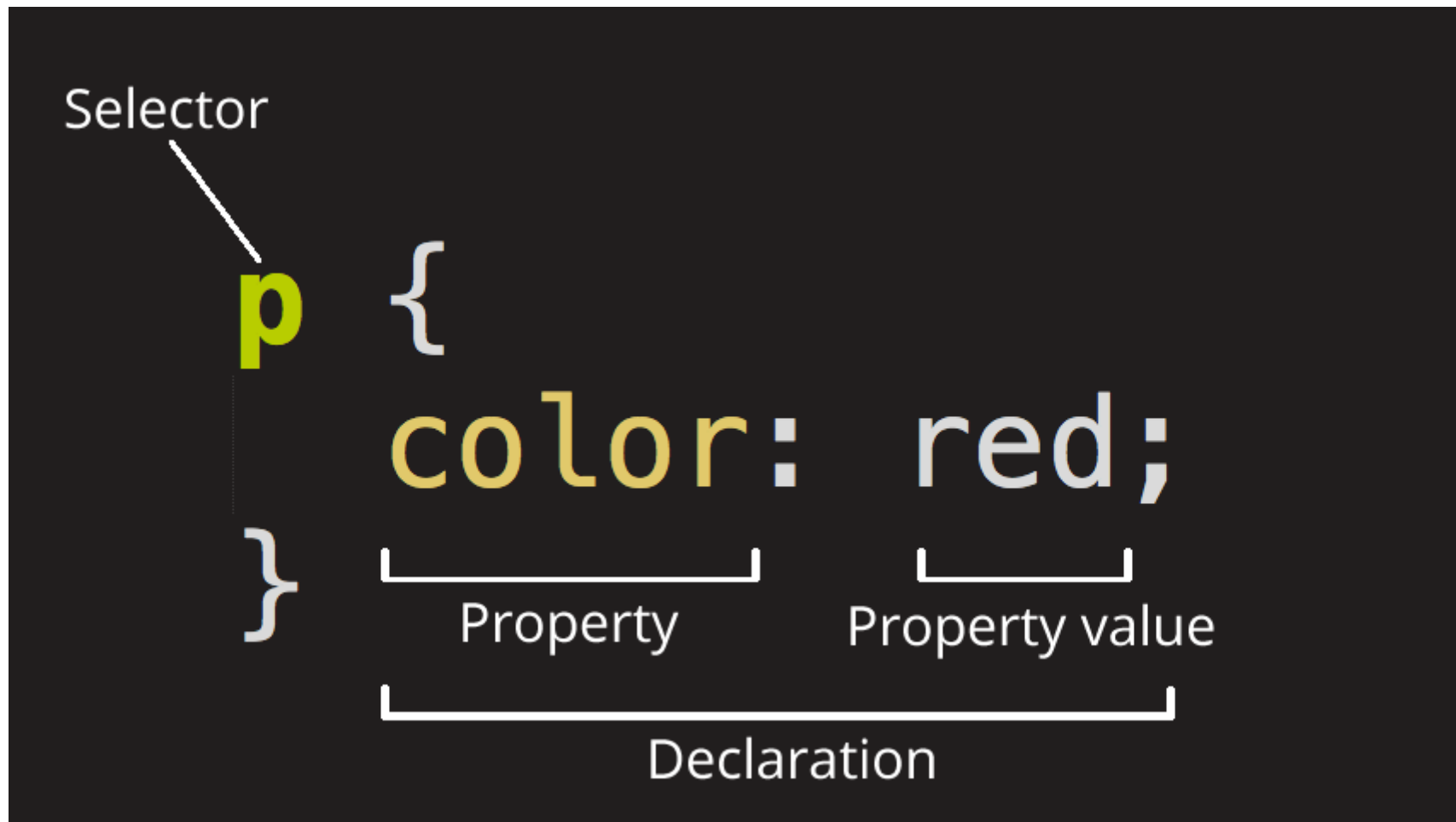




Jika teks paragraf kamu sekarang berwarna merah, selamat! Kamu berhasil menulis CSS pertama kamu.

Bagian-bagian dari sebuah *CSS ruleset*

Mari kita lihat CSS di atas dengan sedikit lebih detail:



Struktur di atas merupakan sebuah **rule set** (atau sering disebut "*rule*" supaya lebih singkat). Perhatikan nama dari masing-masing bagian di atas:

Selector

Nama elemen HTML di awal *rule set*. *Selector* menandai satu atau lebih elemen yang akan diberikan gaya (contoh di sini menggunakan elemen `p`). Untuk memberikan gaya pada elemen lain, kamu bisa mengubah *selector* sesuai elemen yang kamu inginkan.

Declaration

Suatu peraturan seperti `color: red;` yang menentukan properti mana dari elemen yang ingin kamu beri gaya.

Properties

Cara kamu memberikan gaya terhadap elemen HTML (contoh di sini, `color` adalah properti dari {elemen {html{element("p")}}}). Di dalam CSS, kamu memilih properti mana saja yang ingin kamu berikan gaya di peraturan kamu.

Property value

Sebelah sisi kanan properti setelah tanda titik dua, kita memiliki nilai properti, yang memilih satu dari banyak kemungkinan penampilan untuk properti yang diberikan (ada banyak sekali nilai-nilai `color` selain `red`).

Perhatikan juga bagian-bagian lain dari sintaks di atas:

- Masing-masing *rule set* (selain *selector*) harus dibungkus dengan sepasang kurung kurawal (`{ }`).
- Dalam masing-masing deklarasi, kamu harus menggunakan tanda titik dua (`:`) untuk memisahkan properti dari nilai-nilainya.
- Dalam masing-masing *rule set*, kamu harus menggunakan tanda titik-koma (`;`) untuk memisahkan satu deklarasi dengan deklarasi berikutnya.

Untuk memodifikasi banyak nilai properti sekaligus, kamu hanya perlu menggunakan tanda titik-koma sebagai pemisah, seperti ini:

```
1 | p {  
2 |   color: red;
```

```
3 | width: 500px;  
4 | border: 1px solid black;  
5 | }
```

Memilih banyak elemen

Kamu juga dapat memilih lebih dari satu tipe elemen dan mengaplikasikan sebuah *rule set* untuk semua tipe elemen. Untuk melakukannya, kamu dapat menuliskan lebih dari satu *selector* dipisahkan dengan tanda koma. Sebagai contoh:

```
1 | p,li,h1 {  
2 |   color: red;  
3 | }
```

Beragam tipe *selector*

Ada banyak sekali tipe *selector* yang berbeda. Di atas, kita hanya mempelajari **element selectors**, yang memilih seluruh elemen dari tipe yang diberikan di dokumen HTML yang diberikan. Namun kita dapat membuat pilihan yang lebih spesifik daripada itu. Berikut adalah beberapa tipe *selector* lain yang sering dijumpai:

Nama <i>selector</i>	Apa yang dipilih	Contoh
<i>Element selector</i> (kadang-kadang juga disebut sebagai <i>tag</i> atau <i>type selector</i>)	Seluruh elemen HTML dari tipe yang diberikan.	p Memilih <p>

Nama <i>selector</i>	Apa yang dipilih	Contoh
<i>ID selector</i>	Elemen pada halaman dengan ID yang diberikan (pada halaman HTML yang diberikan, kamu hanya boleh memiliki satu elemen per ID).	#my-id Memilih <p id="my-id"> atau
<i>Class selector</i>	Satu atau lebih elemen pada halaman dengan kelas yang ditentukan (banyak <i>class instances</i> yang dapat muncul di sebuah halaman).	.my-class Memilih <p class="my-class"> dan
<i>Attribute selector</i>	Satu atau lebih elemen pada halaman dengan atribut yang ditentukan.	img[src] Memilih namun tidak memilih
<i>Pseudo-class selector</i>	Satu atau lebih elemen yang ditentukan, namun hanya ketika elemen tersebut sedang berada pada <i>state</i> tertentu, seperti sedang di- <i>hover</i> .	a:hover Memilih <a>, namun hanya ketika <i>mouse pointer</i> sedang meng- <i>hover link</i> .

Ada banyak sekali *selector* lain yang dapat dieksplor, dan kamu dapat menemukan daftar yang lebih detil di *Selectors guide* kami.

Fonts dan teks

Setelah mengeksplor dasar-dasar CSS, mari kita tambahkan aturan-aturan lain dan informasi pada *file* `style.css` kita agar contoh yang kita buat memiliki tampilan yang lebih baik. Kita dapat memulai dengan membuat *fonts* dan teks kita untuk terlihat sedikit lebih baik.

1. Pertama-tama, kembalilah dan temukan keluaran dari Google Fonts yang kamu simpan di tempat yang aman. Tambahkan elemen `<link>` di suatu tempat pada bagian *head* dari `index.html` kamu (di manapun di antara *tag* `<head>` dan `</head>`). Elemen *link* tersebut akan terlihat seperti ini:

```
1 | <link href='https://fonts.googleapis.com/css?family=Open+Sans'
```

2. Selanjutnya, hapus aturan yang sudah ada yang kamu miliki di *file* `style.css`.
3. Tambahkan baris berikut, dan ubah baris *placeholder* dengan `font-family` sungguhan yang kamu dapatkan dari Google Fonts. (`font-family` berarti jenis *font* yang ingin kamu gunakan untuk teks.) Aturan ini mengatur jenis *font* dan ukuran *font* dasar secara global untuk seluruh halaman (karena `<html>` merupakan *parent element* dari seluruh halaman, dan semua elemen di dalam halaman tersebut mewarisi `font-size` dan `font-family` yang sama):

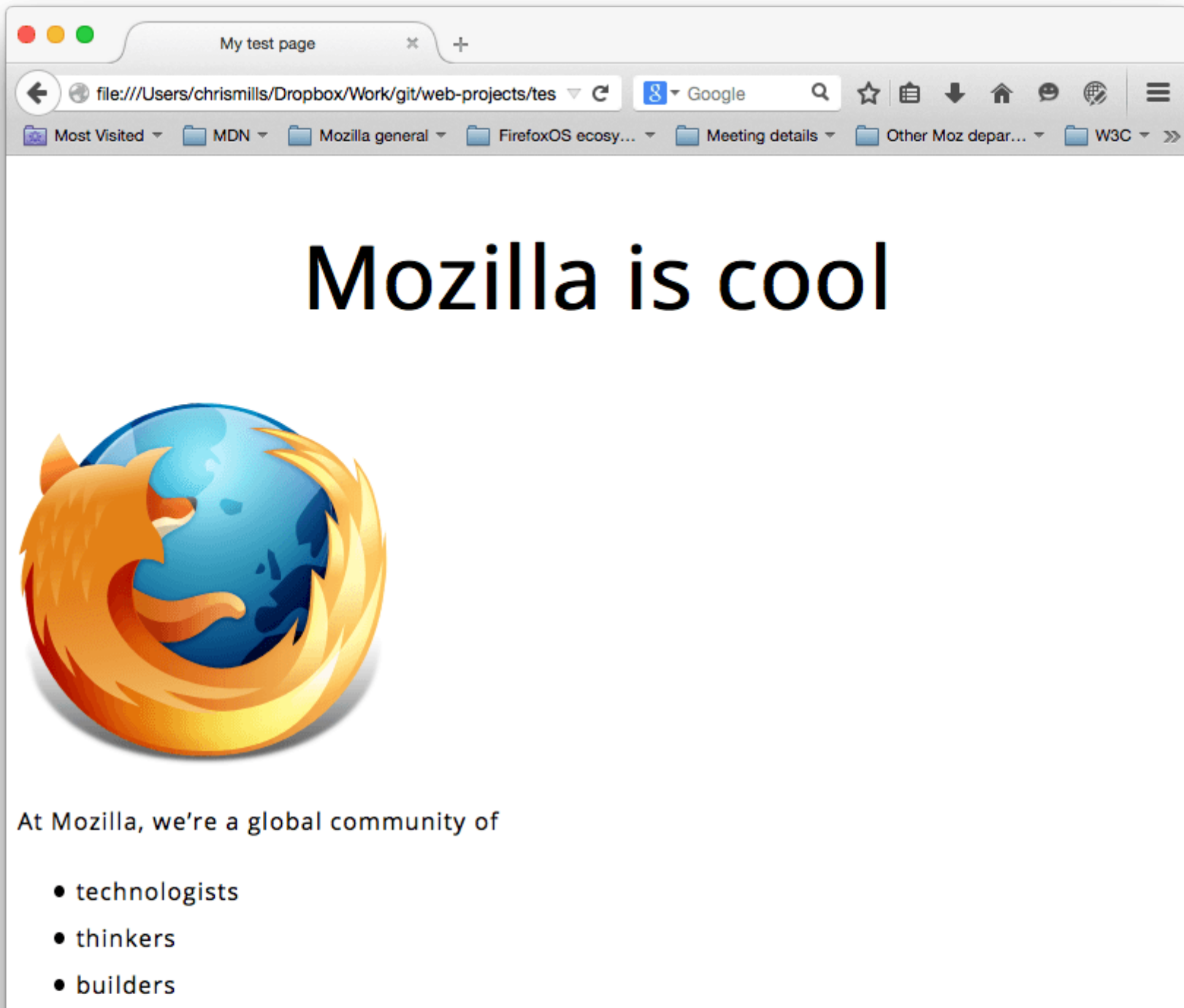
```
1 | html {
2 |     font-size: 10px; /* px means 'pixels': the base font size
3 |     font-family: placeholder: this should be the rest of the
4 | }
```

Catatan: Apapun yang ada di dalam sebuah dokumen CSS antara `/*` dan `*/` adalah **komentar CSS**, yang akan diabaikan oleh *browser* ketika *browser* sedang mengolah kode untuk ditampilkan. Komentar berguna bagi kamu untuk menulis catatan-catatan yang membantu terkait apa yang sedang kamu lakukan.

4. Sekarang kita akan menentukan *font size* untuk elemen-elemen berisi teks yang ada di dalam *body* HTML (`<h1>`, ``, dan `<p>`). Kita juga akan menengahkan posisi teks pada *heading* dan menentukan *line height* serta *letter spacing* pada konten bagian *body* agar dapat lebih mudah dibaca:

```
1  h1 {  
2    font-size: 60px;  
3    text-align: center;  
4  }  
5  
6  p, li {  
7    font-size: 16px;  
8    line-height: 2;  
9    letter-spacing: 1px;  
10 }
```

Kamu dapat menyesuaikan nilai-nilai `px` tersebut untuk mendapatkan desain yang kamu inginkan, namun secara umum desain kamu seharusnya akan terlihat seperti ini:



working together to keep the Internet alive and accessible, so people worldwide can be informed contributors and creators of the Web. We believe this act of human collaboration across an open platform is essential to individual growth and our collective future.

Read the [Mozilla Manifesto](#) to learn even more about the values and principles that guide the pursuit of our mission.

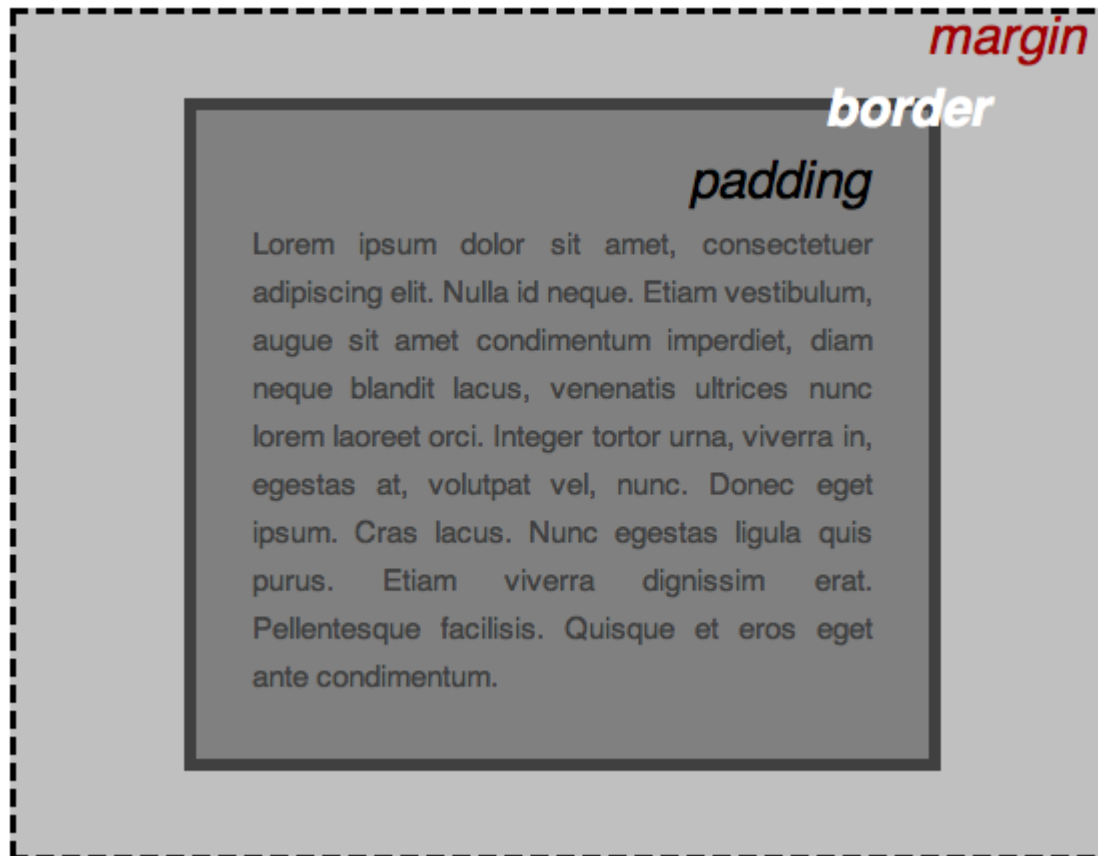
Semua tentang boks

Satu hal yang akan kamu sadari tentang menulis CSS adalah banyak hal dalam CSS sebenarnya terkait dengan boks — mengatur ukuran, warna, posisi, dan lain-lain. Kebanyakan dari elemen-elemen HTML yang ada di halaman kamu dapat dilihat sebagai banyak boks yang saling bertumpuk di atas satu sama lain.



Pada prinsipnya, susunan CSS didasari oleh *box model*. Masing-masing blok akan menempati tempat di halaman kamu dengan properti sebagai berikut:

- *padding*, ruang di sekitar konten (e.g., di sekitar teks paragraf)
- *border*, garis solid yang ada di luar *padding*
- *margin*, ruang di luar elemen



Pada bagian ini kita juga menggunakan:

- `width` (lebar dari sebuah elemen)
- `background-color`, warna di belakang konten dan *padding* sebuah elemen
- `color`, warna dari konten elemen (biasanya berupa teks)
- `text-shadow`: mengatur *drop shadow* pada teks di dalam elemen
- `display`: mengatur tampilan sebuah elemen (untuk sekarang, abaikan hal ini)

Ayo kita mulai menambahkan lebih banyak CSS pada halaman kita! Terus tambahkan aturan-aturan baru sampai ke bagian bawah halaman, dan jangan takut untuk bereksperimen dengan menggonta-ganti nilai-nilai untuk melihat bagaimana hasilnya.

Mengganti warna halaman

```
1 | html {  
2 |     background-color: #00539F;  
3 | }
```

Aturan ini mengatur warna latar belakang seluruh halaman. Gantilah kode warna di atas menjadi warna apapun yang Anda pilih ketika Anda merencanakan situs Anda.

Mengatur *body*

```
1 | body {  
2 |     width: 600px;  
3 |     margin: 0 auto;  
4 |     background-color: #FF9500;  
5 |     padding: 0 20px 20px 20px;  
6 |     border: 5px solid black;  
7 | }
```

Sekarang kita akan mengatur elemen `<body>`. Ada beberapa deklarasi di sini, jadi mari kita bahas semuanya satu per satu:

- `width: 600px;` — aturan ini memaksa *body* untuk selalu memiliki lebar sepanjang 600 piksel.
- `margin: 0 auto;` — ketika kamu mengatur dua nilai pada dua properti seperti `margin` atau `padding`, nilai pertama akan memengaruhi sisi atas dan bawah dari elemen (pada contoh ini, buatlah `0`), dan nilai kedua akan memengaruhi sisi kiri dan kanan (di sini, `auto` adalah sebuah nilai khusus yang dapat membagi ruang horizontal yang ada secara adil antara kiri dan kanan). Kamu juga dapat menggunakan satu, tiga, atau empat nilai seperti yang didokumentasikan di sini.
- `background-color: #FF9500;` — sama seperti sebelumnya, aturan ini menentukan warna latar belakang elemen. Kita sudah menggunakan warna merah kejinggaan untuk bagian *body*, namun teruslah bereksperimen.
- `padding: 0 20px 20px 20px;` — kita memiliki empat nilai yang mengatur *padding* untuk membuat sedikit ruang di sekitar konten kita. Saat ini, kita menentukan tidak ada *padding* pada sisi atas dari *body*, dan ada *padding* sepanjang 20 piksel di sisi kiri, bawah, dan kanan. Nilai-nilai di atas mengatur bagian atas, kanan, bawah, dan kiri secara berurutan.
- `border: 5px solid black;` — aturan ini mengatur *border* setebal 5 piksel berwarna hitam di seluruh sisi *body*.

Mengatur posisi dan memberikan gaya pada judul halaman utama kita

```
1 | h1 {  
2 |     margin: 0;  
3 |     padding: 20px 0;  
4 |     color: #00539F;  
5 | }
```



```
6 | text-shadow: 3px 3px 1px black;  
  | }
```

Kamu mungkin sudah menyadari bahwa ada jarak yang cukup jelek di bagian atas *body*. Hal tersebut terjadi karena beberapa *browser* mengaplikasikan gaya yang bersifat *default* ke elemen `<h1>` element (salah satunya), meski kamu belum memberikan CSS apapun sama sekali! Kedengarannya buruk, namun kita juga menginginkan halaman web yang belum diberikan gaya juga memiliki keterbacaan yang mendasar. Untuk menghilangkan jarak tersebut, kita dapat merubah gaya *default* tersebut dengan mengatur `margin: 0;`.

Selanjutnya, kita mengatur bagian atas dan bawah *padding* menjadi 20 piksel, dan membuat warna dari teks *heading* agar sama dengan warna latar belakang HTML.

Sebuah properti menarik yang kita gunakan adalah `text-shadow`, di mana properti ini akan mengaplikasikan bayangan teks ke konten teks dari sebuah elemen. Empat nilai dari properti tersebut adalah sebagai berikut:

- Nilai piksel pertama mengatur ***horizontal offset*** bayangan dari teks — seberapa jauh bayangan tersebut berada (secara horizontal): sebuah nilai negatif seharusnya akan memindahkan bayangan tersebut ke sisi kiri.
- Nilai piksel kedua mengatur ***vertical offset*** bayangan dari teks — seberapa jauh bayangan tersebut berada (secara vertikal), pada contoh ini, sebuah nilai negatif akan memindahkannya ke atas.
- Nilai piksel ketiga mengatur ***blur radius*** dari bayangan — nilai yang lebih besar akan membuat bayangan yang lebih kabur.
- Nilai keempat mengatur warna dasar dari bayangan.

Cobalah untuk bereksperimen dengan berbagai nilai berbeda dan lihat hasilnya!

Memposisikan gambar ke tengah

```
1 | img {  
2 |   display: block;  
3 |   margin: 0 auto;  
4 | }
```

Akhirnya, kita akan memposisikan gambar ke tengah untuk membuatnya terlihat lebih baik. Kita dapat menggunakan trik `margin: 0 auto` lagi seperti yang telah kita lakukan pada bagian *body*, namun ada satu hal lagi yang perlu dilakukan. Elemen `<body>` adalah **block level**, di mana elemen tersebut menempati ruang di halaman dan kita juga dapat mengaplikasikan margin dan nilai-nilai *spacing* lain pada elemen tersebut. Di sisi lain, gambar adalah contoh dari *inline element*, yang berarti kita tidak dapat melakukan hal yang sama. Sehingga untuk mengaplikasikan *margin* pada gambar, kita harus memberikan gambar tersebut perlakuan seperti *block level* menggunakan `display: block;`.

Catatan: Jangan khawatir jika kamu belum memahami `display: block;` dan perbedaan antara *block-level* dan *inline*. Kamu akan memahaminya seiring dengan waktu kamu mempelajari CSS lebih dalam. Kamu dapat menemukan lebih lanjut tentang nilai-nilai *display* yang tersedia di halaman referensi *display* kami.

Kesimpulan

Jika kamu sudah mengikuti seluruh instruksi di artikel ini, seharusnya kamu memiliki sebuah halaman yang tampak seperti ini (kamu juga dapat melihat versi kami di sini):



worldwide can be informed contributors and creators of the Web. We believe this act of human collaboration across an open platform is essential to individual growth and our collective future.

Read the [Mozilla Manifesto](#) to learn even more about the values and principles that guide the pursuit of our mission.

Jika Kamu mengalami kebingungan, kamu bisa membandingkan hasilmu dengan contoh sampel kode kami di Github.

Di sini, kita hanya baru saja membahas sedikit tentang CSS. Untuk mempelajari lebih lanjut, kunjungi halaman *CSS Learning topic* milik kami.
