**Rangkuman Pendahuluan**

Kita telah belajar banyak hal dalam modul Pendahuluan. Kami siapkan rangkuman materi terhadap hal yang telah Anda pelajari selama ini. Harapannya, Anda dapat mengingat kembali setiap materi yang disampaikan.

**Cara Internet Bekerja**

Internet memberikan kemampuan pada user untuk mengirimkan permintaan dan menerima hasil permintaan (data) pada website. Hal ini terjadi karena komputer milik user terhubung dengan komputer lain yang biasa disebut dengan server. Kedua komputer ini berkomunikasi melalui sebuah jaringan, baik dengan kabel maupun nirkabel.

Berikut adalah perkembangan jaringan yang terjadi.

| **Jaringan** | **Keterangan** |
| --- | --- |
| [A diagram of a line between two circles  Description automatically generated with medium confidence](https://www.dicoding.com/academies/123/tutorials/32105) | Dua buah komputer saling terhubung untuk berkomunikasi, baik menggunakan kabel atau nirkabel. |
| [A diagram of a network  Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32105) | Banyak komputer terhubung dengan banyak komputer juga. |
| [A diagram of a network  Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32105) | Jaringan ini sudah terbentuk dengan memanfaatkan network switch agar dapat terkoneksi dengan mudah dan efisien. Bahkan, satu jaringan dengan network switch dapat terhubung dengan network switch lain sehingga memperbesar koneksi komputer. |
|  | Suatu jaringan dapat terhubung dengan jaringan lainnya meskipun berbeda kontinen. Ia menggunakan Internet Service Provider (ISP) agar saling terhubung dengan router ISP lainnya. |

**Website dan Halaman Web**

Berikut adalah penjelasan singkat mengenai halaman web dan website.

**Halaman Web**

[A screenshot of a web page

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32105)

Halaman web merupakan halaman tunggal yang menampilkan informasi, baik berupa teks, gambar, maupun media lainnya. Tidak hanya media yang ditampilkan, halaman web yang dibangun dapat ditambahkan interaktivitas menggunakan kode (JavaScript).

**Website**

[](https://www.dicoding.com/academies/123/tutorials/32105)

Jika sebelumnya Anda belajar halaman web berupa halaman tunggal, website adalah sekumpulan halaman web yang saling terhubung. Website bersifat unik karena setiap halaman dapat saling terhubung dengan menggunakan hipertaut (*hyperlink*), baik dalam domain yang sama maupun berbeda.

**Web Server**

Mari mengetahui web server lebih dalam.

**Siklus Request dan Response**

Tahukah Anda bahwa browser dapat menampilkan website dengan baik karena mendapatkan data dari komputer lain yang biasa disebut dengan server? Bagi komputer yang mengakses website, kita disebut sebagai *client*, atau dalam hal ini adalah browser yang melakukan permintaan data.

Browser akan mengirimkan sesuatu bernama *request* pada server dan menerima data dalam sesuatu bernama *response* sebagai hasil tanggapan dari server. Data-data tersebut dapat berupa berkas HTML, CSS, JavaScript, dan aset-aset lain yang dibutuhkan untuk menampilkan website.

[A black line with a white background

Description automatically generated with medium confidence](https://www.dicoding.com/academies/123/tutorials/32105)

**Peranan Web Server**

Berbicara mengenai web server, ia dapat terbagi menjadi dua hal, yaitu *hardware* dan *software*. Bukan berarti kedua hal tersebut bekerja secara terpisah, tetapi saling melengkapi dan bekerja sama.

Dari sisi hardware, web server merupakan komputer dengan spesifikasi yang disesuaikan berdasarkan layanannya. Contohnya, kapasitas hard drive yang besar akan dibutuhkan jika Anda memiliki website yang menyimpan banyak gambar.

Dari sisi software, web server merupakan komputer yang menjalankan sebuah program agar dapat melayani (menerima atau mengirim) data melalui jenis protokol bernama HTTP. Ini merupakan protokol standar dalam melakukan transaksi data oleh browser.

[A diagram of a diagram of a computer

Description automatically generated with medium confidence](https://www.dicoding.com/academies/123/tutorials/32105)

**DNS Server**

Sebenarnya, nama domain tidak akan menggantikan peran dari IP address. Komputer tetap menggunakan alamat IP untuk mengakses website. Namun, bagaimana caranya nama domain dapat diterjemahkan menjadi alamat IP? Berikut jawabannya.

Ketika user membuka website Dicoding menggunakan domain dicoding.com, browser akan menanyakan pada komputer, apakah ia mengenali dan dapat memberikan IP address-nya berdasarkan nama domain tersebut? Dia akan memeriksa berdasarkan DNS cache yang ada. Jika ada, browser akan diberikan alamat IP-nya dan menampilkan website yang diminta.

Jika komputer tidak mengenali nama domain tersebut, dia akan menanyakan pada DNS server yang secara singkat bertugas untuk memberi tahu alamat IP yang sesuai dari nama domain yang terdaftar padanya. Jika sudah, browser akan diberikan alamat IP-nya dan meneruskan permintaan konten halaman web ke web server.

[A diagram of a computer server

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32105)

**Web Browser**

Web browser atau disingkat browser merupakan sebuah program yang menampilkan halaman web. Tidak hanya satu halaman web, tetapi banyak website yang aktif di dunia ini dapat diakses oleh browser. Bahkan, halaman web yang saling terhubung dalam satu domain yang sama juga dapat diakses.

**Cara Browser Bekerja**

Ketika user mengakses website dengan URL, browser akan membuat sebuah request yang akan dikirimkan ke web server. Kemudian, web server akan mengirimkan response berisi konten-konten yang dibutuhkan oleh website dan browser akan memprosesnya hingga tampil website pada perangkat user.

Browser memiliki address bar yang merupakan sebuah input bagi user untuk menuliskan alamat website atau yang biasa disebut Uniform Resource Locator (URL). Ada tiga bagian pada URL, yaitu protocol (https), domain name (dicoding.com), dan file path (alamat halaman web yang sedang ditampilkan).

[A screen shot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32105)

**Anatomi Website**

Pada dasarnya, website dibangun dengan tiga buah fondasi utama:

* HTML: sebuah *markup language* yang menentukan struktur atau kerangka dari halaman web;
* CSS: sebuah *markup language*yang memberikan gaya atau styling agar halaman web ditampilkan lebih ciamik; dan
* JavaScript: satu-satunya *programming language* yang didukung oleh browser untuk meningkatkan interaktivitas pada halaman web.

Perlu ditekankan bahwa ketiga komponen tersebut termasuk hal vital dalam pembangunan web. Sebenarnya, kita bisa saja membangun halaman web tanpa CSS dan JavaScript. Namun, website akan terlihat tidak menarik dan membosankan.

Untuk lebih memudahkan penggambaran, berikut adalah analogi dari penerapan tiga fondasi website.

[A group of cats with text

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32105)

**Rangkuman Pengenalan HTML**

Kita telah belajar banyak hal dalam modul Pengenalan HTML. Kami siapkan rangkuman materi terhadap hal yang telah Anda pelajari selama ini. Harapannya, Anda dapat mengingat kembali setiap materi yang disampaikan.

**Pengantar HTML**

[A logo of a website

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32110)

Apakah Anda masih ingat HTML? Ia merupakan satu-satunya bahasa markup untuk membangun struktur halaman website. Sebagaimana yang telah dipelajari sebelumnya, HTML analogikan seperti kerangka hewan yang menentukan bentuk tubuhnya. Hal ini menjadikan setiap hewan memiliki bentuk yang bermacam-macam.

**Anatomi Elemen HTML**

Elemen HTML adalah salah satu bagian dari HTML dalam membangun halaman web. Elemen ini digunakan untuk mendefinisikan elemen-elemen yang ditampilkan dalam halaman web. Ada beberapa hal untuk membangun elemen HTML itu sendiri. Secara garis besar, berikut adalah bagan dari anatomi elemen HTML.

[A black rectangular sign with white text

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32110)

Berikut adalah pembahasan dari masing-masing bagian dari bagan di atas.

| **Item** | **Keterangan** |
| --- | --- |
| Tag pembuka | Berisi nama tag dari elemen yang akan dibuat dan dibungkus dengan angle bracket (<>). Contohnya adalah **<p>** untuk membuat elemen paragraf yang menggunakan **p** sebagai nama elemennya--singkatan dari dalam bahasa inggris (paragraph). |
| Konten | Konten dari elemen. Contohnya teks sebagai konten dari elemen paragraf. |
| Tag penutup | Masih sama seperti tag pembuka, kecuali terdapat garis miring sebelum nama elemennya. Ini menandakan akhir dari elemen HTML. Biasanya, kesalahan bagi web programmer pemula adalah melupakan tag ini sehingga menyebabkan elemen tidak valid. |

**Attribute di Elemen HTML**

Dalam membuat elemen HTML, ada satu hal yang dapat dilakukan, yaitu memberi atribut. Atribut dapat memberi informasi-informasi tambahan untuk elemen HTML. Informasi ini tidak akan tampil dalam halaman web, tetapi ia dapat menentukan perilaku elemen biasanya. Berikut adalah anatomi dari atribut elemen untuk memperjelas pemahaman Anda.

[A black rectangle with white text

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32110)

**Anatomi Dokumen HTML**

Pada dasarnya, dokumen HTML memerlukan struktur dasar untuk menampilkan halaman web dengan baik. Halaman web seharusnya memiliki susunan elemen HTML yang tampak seperti berikut.

[A computer screen shot of text

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32110)

Dokumen di atas sebetulnya akan membentuk sebuah hierarki elemen atau yang biasa disebut dengan DOM Tree (pohon DOM). Ini dapat Anda analogikan seperti silsilah keluarga. Kurang lebih, berikut adalah DOM Tree yang terbentuk dari dokumen HTML di atas.

[A diagram of a website

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32110)

**Rangkuman Pendalaman HTML**

Kita telah belajar banyak hal dalam modul Pendalaman HTML. Kami siapkan rangkuman materi terhadap hal yang telah Anda pelajari selama ini. Harapannya, Anda dapat mengingat kembali setiap materi yang disampaikan.

**Paragraf**

Paragraf adalah elemen paling mendasar dari sebuah dokumen teks. Pada HTML, kita bisa menunjukkan sebuah paragraf dengan menggunakan elemen <p>. Contohnya berikut.

1. <p>Ini adalah teks dalam elemen paragraf</p>

**Heading**

Dalam menuliskan struktur konten artikel, kita memerlukan judul sebagai topik pembahasan, paragraf sebagai penjelas dari topik yang dibawa, hingga subjudul lagi di dalamnya yang merupakan topik lebih rinci dari topik utama beserta penjelasannya (paragraf), dan seterusnya. Pada HTML, terdapat <h1> hingga <h6> elemen heading yang dapat kita gunakan.

[A screenshot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32135)

**List**

Dalam membuat sebuah daftar barang, kita dapat menampilkannya dalam bentuk list. HTML memiliki elemen khusus untuk hal tersebut.

**Unordered List**

Unordered list merupakan daftar yang tidak mementingkan urutan. Standarnya, unordered list menampilkan bullet pada tiap item list-nya (tetapi kita bisa mengubahnya dengan styling).

[A screenshot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32135)

**Ordered List**

*Ordered list* digunakan untuk membuat list yang mementingkan urutan. Contohnya, membuat daftar instruksi langkah demi langkah sehingga dibutuhkan urutan yang sesuai. Ordered list bekerja seperti unordered list, tetapi perbedaannya adalah pada tiap item menampilkan angka bukan sebuah *bullet*.

[A screenshot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32135)

**Gambar**

Pada HTML untuk menampilkan sebuah gambar kita bisa menggunakan tag <img>. Berbeda dengan elemen lain, elemen <img> tidak menuliskan konten di antara tag pembuka dan tag penutup. Namun, untuk menetapkan gambar yang ditampilkan kita gunakan sebuah atribut**.**

* [**HTML**](https://www.dicoding.com/academies/123/tutorials/32135#tab1-code1)
* [Hasil](https://www.dicoding.com/academies/123/tutorials/32135#tab1-code2)

1. <img src="https://raw.githubusercontent.com/dicodingacademy/BelajarDasarPemrogramanWeb/099-shared-files/dicoding-logo.png" alt="dicoding">

Hal yang perlu kita perhatikan adalah atribut pada elemen tersebut, terdapat dua elemen yang harus kita gunakan ketika menerapkan elemen <img>.

* **src**  
  Atribut ini berfungsi sebagai sumber dari gambar yang ditampilkan. Atribut ini dapat bernilai url gambar atau path gambar local dari gambar yang digunakan.
* **alt**  
  Atribut ini sebenarnya tidak wajib untuk diterapkan, hanya saja atribut ini akan sangat berguna ketika gambar tidak berhasil ditampilkan. Nilai atribut ini merupakan tampilan dari gambar yang ditampilkan dalam bentuk tulisan.

**Inline Formatting Text: Long Quotations, Preformatted Text, dan Figure**

Sejauh ini, kita sudah mengenal paragraf, heading, dan juga list pada HTML. Namun, masih ada beberapa lagi yang merupakan spesial teks format yang dapat kita gunakan yaitu <blockquote>, <pre> dan <figure>.

**Long Quotations**

Jika pada konten kita memiliki sebuah kutipan ataupun sebuah testimonial, kita dapat gunakan format *long quotations* dengan menggunakan tags <blockquote>. Konten di dalam elemen <blockquote> dapat berupa sebuah paragraf, heading, ataupun list.

[A screenshot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32135)

Pada elemen ini, kita dapat menggunakan atribut *cite* untuk menentukan sumber URL dari sebuah kutipan (jika kutipan tersebut bersumber dari sebuah situs website). Namun, tidak ada tampilan yang berbeda pada browser secara kasat mata.

**Preformatted Text**

HTML akan mengabaikan spasi yang dituliskan secara berulang dan juga *line breaks* (garis baru). Namun, pada beberapa tipe konten, seperti contoh kode atau puisi hal tersebut sangat berarti. Ada sebuah elemen yang dapat kita gunakan untuk menampilkan konten sesuai yang kita tulis pada text editor. Untuk menggunakannya, kita gunakan elemen <pre> sebagai pembungkus kontennya.

* [**HTML**](https://www.dicoding.com/academies/123/tutorials/32135#tab2-code1)
* [Hasil](https://www.dicoding.com/academies/123/tutorials/32135#tab2-code2)

1. <pre>
2. SAJAK PUTIH
4. Bersandar pada tari warna pelangi
5. Kau depanku bertudung sutra senja
6. Di hitam matamu kembang mawar dan melati
7. Harum rambutmu mengalun bergelut senda
9. Sepi menyanyi, malam dalam mendoa tiba
10. Meriak muka air kolam jiwa
11. Dan dalam dadaku memerdu lagu
12. Menarik menari seluruh aku
14. Hidup dari hidupku, pintu terbuka
15. Selama matamu bagiku menengadah
16. Selama kau darah mengalir dari luka
17. Antara kita Mati datang tidak membelah...
19. Karya: Chairil Anwar
20. </pre>

**Figure**

Elemen <figure> digunakan untuk mempresentasikan konten tersendiri (*self-contained content*), seperti ilustrasi, diagram, foto, atau bisa juga sebuah baris kode. Dalam elemen figure, kita dapat menuliskan elemen <figcaption> sebagai sebuah caption (judul) untuk konten tersebut.

Berikut contoh penerapan elemen ini untuk menampilkan gambar.

[A screenshot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32135)

**Inline Formatting Text: Anchor, Emphasized Text, Important Text, dan Short Quotation**

Pada standarnya, elemen HTML memiliki dua sifat, yaitu block dan inline. Elemen yang bersifat block selalu membuat baris baru ketika di-render. Contohnya seperti elemen paragraf, list, heading, dan lainnya. Lawan dari elemen tersebut, yaitu elemen inline. Elemen ini tidak menambahkan baris baru ketika di-render. Apa saja elemen tersebut? Mari kita bahas satu persatu.

**Anchor**

Anchor (jangkar) merupakan elemen yang digunakan untuk membuat sebuah hyperlink ke halaman atau website lain, file, alamat email, atau URL lainnya. Selain itu, ada atribut wajib agar elemen ini berfungsi dengan baik, yaitu href untuk menetapkan sebuah target yang dituju.

* [**HTML**](https://www.dicoding.com/academies/123/tutorials/32135#tab3-code1)
* [Hasil](https://www.dicoding.com/academies/123/tutorials/32135#tab3-code2)

1. <p>Hubungi kami di</p>
2. <ul>
3. <li><a href="https://example.com">Website</a></li>
4. <li><a href="mailto:info@example.com">Email</a></li>
5. <li><a href="tel:+62123456">Telepon</a></li>
6. <li><a href="#address">Alamat</a></li>
7. </ul>

**Emphasized Text**

Gunakan elemen <em> untuk menunjukkan bagian kata yang perlu kita tekankan. Elemen ini menunjukkan *stress emphasis*atau konten/kata yang perlu mendapatkan penekanan atau perhatian khusus.

* [**HTML**](https://www.dicoding.com/academies/123/tutorials/32135#tab4-code1)
* [Hasil](https://www.dicoding.com/academies/123/tutorials/32135#tab4-code2)

1. <p><em>Dia</em> adalah seorang pelajar</p>
2. <p>Dia adalah seorang <em>pelajar</em></p>

**Important Text**

Gunakan elemen <strong> untuk menunjukkan sebuah teks yang begitu penting (*strong importance*), serius ataupun mendesak. Artinya, teks tersebut harus dapat perhatian lebih dari teks biasa lainnya.

* [**HTML**](https://www.dicoding.com/academies/123/tutorials/32135#tab5-code1)
* [Hasil](https://www.dicoding.com/academies/123/tutorials/32135#tab5-code2)

1. <p>Kesehatan merupakan hal yang penting, jaga pola makan yang teratur dan <strong>jangan sampai makan tengah malam!</strong></p>

**Short Quotations**

Gunakan elemen <q> untuk menandai sebuah kutipan dalam sebuah teks. Elemen ini berbeda dengan <blockquote>. Elemen ini digunakan untuk kutipan pendek yang terletak di dalam baris (*inline*).

* [**HTML**](https://www.dicoding.com/academies/123/tutorials/32135#tab6-code1)
* [Hasil](https://www.dicoding.com/academies/123/tutorials/32135#tab6-code2)

<p>Sebelum pulang kerja, ia berkata kepadaku: <q>Maaf saya tidak bisa hadir dalam pertemuan nanti</q></p>

Sama seperti elemen blockquote, elemen quotation marks memiliki atribut **cite** untuk menentukan sumber URL dari sebuah kutipan (jika kutipan tersebut bersumber dari sebuah situs website). Namun, tidak ada perbedaan yang terlihat secara kasat mata jika dijalankan di browser.

**Inline Formatting Text: Citation, Defining Terms, Subscript, Superscript, Highlighted Text, dan Line Break**

Kita sudah belajar banyak formatting text. Namun, ada beberapa sisanya yang belum kita bahas. Tentunya, mereka tidak kalah penting dalam membangun konten halaman web yang lebih baik. Mari kita lengkapi pembahasan ini. *Jom!*

**Citation**

Selain sebuah atribut, <cite> juga merupakan sebuah elemen yang digunakan untuk sebuah rujukan pada sebuah dokumen, contohnya sebuah buku, majalah, artikel, dan lainnya.

1. <p>Informasi selengkapnya bisa Anda dapatkan di <cite><a href="https://dicoding.com">dicoding.com</a></cite>.</p>

Standarnya, pada browser sebuah teks yang diberi markup <cite> akan ditampilkan dengan huruf miring (*italic*).

**Defining Terms**

Elemen <dfn> digunakan ketika mendefinisikan sebuah istilah (*term*). Elemen ini harus terletak pada elemen lain yang menaunginya. Contohnya pada sebuah elemen <p> atau elemen <section>.

1. <p><dfn>Website</dfn> merupakan halaman yang menampilkan informasi melalui teks atau gambar. Website dapat diakses melalui internet dengan menggunakan browser.</p>

Standar pada browser yakni sebuah teks yang diberi markup <dfn> akan ditampilkan dengan huruf miring (*italic*).

**Subscript dan Superscript**

Subscript <sub> dan superscript <sup> adalah elemen yang dapat membuat teks yang ditampilkan tampak kecil, dengan posisi di bawah (sub) atau di atas (sup) dari teks biasanya. Elemen ini digunakan untuk menunjukkan sebuah rumus kimia ataupun matematika.

1. <p>
2. Sukrosa merupakan suatu disakarida yang dibentuk dari monomer-monomernya yang berupa unit glukosa dan fruktosa, dengan rumus molekul C<sub>12</sub>H<sub>22</sub>O<sub>11</sub>.
3. </p>
5. <p>Salah satu persamaan paling umum dalam semua fisika adalah E=MC<sup>2</sup></p>

**Highlighted Text**

Untuk menandai atau menyorot sebuah teks kita bisa menggunakan elemen <mark>. Elemen ini digunakan ketika terdapat sebuah teks yang memiliki peran penting. Biasanya teks tersebut merupakan bagian yang paling relevan atau penting dalam sebuah konteks kalimat.

1. <p>
2. Ini adalah periode perang saudara. Pesawat ruang angkasa pemberontak, menyerang dari pangkalan
3. tersembunyi, telah memenangkan kemenangan pertama mereka melawan Kekaisaran Galactic yang jahat.
4. Selama pertempuran,
5. <mark>mata-mata Pemberontak berhasil mencuri rencana rahasia </mark>
6. ke senjata pamungkas Kekaisaran, STAR DEATH, stasiun ruang angkasa berlapis baja dengan kekuatan
7. yang cukup untuk menghancurkan seluruh planet.
8. </p>

Standarnya, pada browser teks yang diberi markup <mark> akan ditampilkan dengan background kuning dan teks hitam.

**Line Break**

*Inline line break element* (<br>) dapat digunakan untuk memberitahu browser untuk memberikan sebuah garis baru pada baris teks. Sama seperti gambar, elemen ini merupakan elemen kosong sehingga kita tidak membutuhkan sebuah tag penutup.

1. <p>
2. Dicoding Space,<br>
3. Jln. Batik Kumeli No. 50.<br>
4. Bandung.<br>
5. 40123
6. </p>

**Semantic HTML: Mengorganisasikan Halaman Konten**

Website memiliki hierarki konten yang sama seperti dokumen sehari-hari yang kita baca, majalah, dan koran contohnya. Jadi, hierarki pada sebuah website merupakan hal yang penting. Tentu elemen yang terdapat pada HTML perlu kita kelompokkan menjadi beberapa bagian.  
[A blank rectangular white paper

Description automatically generated with low confidence](https://www.dicoding.com/academies/123/tutorials/32135)

Kita dapat menggunakan beberapa elemen dalam HTML untuk mengelompokkan sebuah elemen dengan jelas dan memiliki arti (*semantic meaning*). Elemen-elemen ini memiliki nama sesuai dengan fungsi atau peran dari elemen tersebut.

[A close-up of a check

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32135)

**Semantic HTML: Header, Footer, Main, dan Nav**

Sebelumnya, kita sudah melihat gambar yang mengelompokkan konten halaman web dengan terstruktur. Organisasi konten tersebut mirip seperti informasi di media cetak seperti koran, bukan? Nah, mari kita belajar beberapa elemen yang berperilaku seperti konten kepala, pokok, samping, dan kaki.

**Header dan Footer**

Sebuah *header* dan *footer* utama yang muncul pada awal dan akhir di sebuah halaman (<body>). Header sebagai pengantar atau pembuka konten dalam sebuah elemen <article> atau <section>, sedangkan Footer sebagai catatan kaki pada sebuah elemen <article> atau <section>.

Pada contoh di bawah ini, elemen <header> digunakan untuk menampilkan nama situs dan navigasi utama serta elemen <footer> digunakan sebagai informasi hak cipta.

* [**<header>**](https://www.dicoding.com/academies/123/tutorials/32135#tab7-code1)
* [<footer>](https://www.dicoding.com/academies/123/tutorials/32135#tab7-code2)

1. <header>
2. <h1>Bandung</h1>
3. <p>
4. Kota metropolitan terbesar di Provinsi Jawa Barat, sekaligus menjadi ibu kota provinsi tersebut.
5. </p>
6. <nav>
7. <ul>
8. <li><a href="#sejarah">Sejarah</a></li>
9. <li><a href="#geografis">Geografis</a></li>
10. <li><a href="#wisata">Wisata</a></li>
11. </ul>
12. </nav>
13. </header>

**Main**

Element **<main>** digunakan untuk menampung/mewadahi konten utama (dominan) dalam **<body>**. Konten main dapat terdiri dari banyak section, ataupun artikel, atau konten apa pun di dalam elemen main, selama ia termasuk konten utama yang dimiliki oleh website.

1. <body>
2. <header>
3. <!-- Konten header website -->
4. </header>
6. **<main>**
7. **<!-- Konten utama website -->**
8. **</main>**
10. <footer>
11. <!-- Konten footer website -->
12. </footer>
13. </body>

**Nav**

Elemen **<nav>** digunakan untuk menampung sebuah navigasi yang sifatnya penting (mayor), contohnya navigasi utama pada sebuah website.

1. <nav>
2. <ul>
3. <li><a href="#sejarah">Sejarah</a></li>
4. <li><a href="#geografis">Geografis</a></li>
5. <li><a href="#wisata">Wisata</a></li>
6. </ul>
7. </nav>

Sebuah navigation pada dasarnya sangat berguna untuk aksesibilitas website kita. Contohnya ketika pengguna website kita menggunakan *screen reader* dalam mengunjungi website, pengguna akan mudah mencari bagian yang dia inginkan tanpa harus menelusuri seluruh konten website.

**Semantic HTML: Articles, Aside, dan Section**

Sebelumnya, kita telah membahas beberapa elemen yang bisa dikatakan merupakan konten besar. Hal tersebut karena konten-konten tersebut terlihat jelas pembagiannya. Ada beberapa semantik HTML lainnya yang lebih kecil dan spesifik tentunya. Kali ini, kita akan membahasnya.

**Articles**

Elemen <article> bertindak sebagai *container* untuk *independent content* pada sebuah halaman, artinya konten utuh yang tidak terkait dengan konten lain, bisa saja sebuah artikel blog, komentar, forum post dan konten lainnya.

1. <article>
2. <h2>Sejarah</h2>
3. <img src="img/history.jpg">
4. <p>Kata Bandung berasal dari kata bendung atau bendungan karena terbendungnya sungai Citarum oleh lava Gunung Tangkuban Parahu yang lalu membentuk telaga...</p>
5. </article>
7. <article>
8. <h2>Geografis</h2>
9. <img src="img/geografis.jpg">
10. <p>Kota Bandung dikelilingi oleh pegunungan, sehingga bentuk morfologi wilayahnya bagaikan sebuah mangkok raksasa,[9] secara geografis kota ini terletak di tengah-tengah provinsi Jawa Barat..</p>
11. </article>
13. <article>
14. <h2>Wisata</h2>
15. <img src="img/farm-house.jpg">
16. <p>Sejak dibukanya Jalan Tol Cipularang, kota Bandung telah menjadi tujuan utama dalam menikmati liburan akhir pekan terutama dari masyarakat yang berasal dari Jakarta sekitarnya...</p>
17. </article>

Elemen <article> dapat berada pada elemen <article> lainnya (nested) selama artikel tersebut berkaitan dengan induk elemen <article> yang menampungnya.

**Aside**

Elemen <aside> memiliki dua tujuan, tergantung kita menempatkannya di dalam sebuah elemen <article> atau tidak.

Ketika elemen ini ditempatkan di dalam elemen <article>, elemen ini dapat berisi informasi yang berhubungan dengan artikel tersebut, tetapi bukan bagian dari konten artikelnya itu sendiri (dipisahkan dari konten utama).

Ketika elemen ini ditempatkan di luar elemen <article>, elemen ini dapat berisi informasi yang berhubungan pada keseluruhan halaman.

Elemen <aside> biasanya terletak di samping dari sebuah elemen yang menampungnya.

1. <main>
2. <div id="content">
3. <!-- ...... -->
4. </div>
6. <aside>
7. <article>
8. <header>
9. <h2>Kota Bandung</h2>
10. <p>Kota Kembang Paris van Java</p>
11. <figure>
12. <img src="img/bandung-badge.png" />
13. <figcaption>Lambang</figcaption>
14. </figure>
15. </header>
16. </article>
17. </aside>
18. </main>

**Section**

Sebuah elemen yang memiliki kesamaan konten dan sebuah *heading* di dalamnya dapat dikelompokkan dengan menggunakan elemen <section>. Dengan begitu elemen ini dapat digunakan pada sebuah elemen <article> yang memiliki konten panjang dan berpotensi untuk dikelompokkan.

1. <article>
2. <section>
3. <h2>Wisata</h2>
4. <p>
5. Sejak dibukanya Jalan Tol Cipularang, kota Bandung telah menjadi tujuan utama dalam menikmati
6. liburan akhir pekan terutama dari masyarakat yang berasal dari Jakarta sekitarnya. Selain
7. menjadi kota wisata belanja, kota Bandung juga dikenal dengan sejumlah besar bangunan lama
8. berarsitektur peninggalan Belanda.
9. </p>
10. </section>
12. <section>
13. <h3>Farm House Lembang</h3>
14. <img src="img/farm-house.jpg" />
15. <p>
16. Kota Bandung dikelilingi oleh pegunungan, sehingga bentuk morfologi wilayahnya bagaikan sebuah
17. mangkok raksasa,[9] secara geografis kota ini terletak di tengah-tengah provinsi Jawa Barat,
18. serta berada pada ketinggian ±768 m di atas permukaan laut, dengan titik tertinggi di berada
19. di sebelah utara dengan ketinggian 1.050 meter di atas permukaan laut dan sebelah selatan
20. merupakan kawasan rendah dengan ketinggian 675 meter di atas permukaan laut.
21. </p>
22. </section>
24. <section>
25. <h3>Observatorium Bosscha</h3>
26. <img src="img/bosscha.jpg" />
27. <p>
28. Memiliki beberapa teleskop, antara lain, Refraktor Ganda Zeiss, Schmidt Bimasakti, Refraktor
29. Bamberg, Cassegrain GOTO, dan Teleskop Surya. Refraktor Ganda Zeiss adalah jenis teleskop
30. terbesar untuk meneropong bintang. Benda ini diletakkan pada atap kubah sehingga saat teropong
31. digunakan, atap tersebut harus dibuka. Observatorium Bosscha boleh dikunjungi oleh siapa pun,
32. tanpa tiket. Namun, bagi yang ingin menggunakan teleskop Zeiss, wajib mendaftarkan diri. Untuk
33. instansi atau lembaga pendidikan, diberikan jadwal hari Selasa sampai Jumat. Sementara itu,
34. kunjungan individu dibuka setiap hari Sabtu.
35. </p>
36. </section>
37. </article>

**Generic Element**

HTML menyediakan dua tipe elemen umum (*generic element*) yang bisa kita kustomisasi untuk menggambarkan konten kita dengan tepat, yaitu div dan span. Elemen ini akan terlibat jika tidak ada semantic element sesuai di HTML.

**Div**

Elemen <div> merupakan sebuah wadah (container) yang bersifat umum untuk menampung beberapa konten. Elemen ini tidak akan memberikan efek apa pun pada konten atau layout sebelum menerapkan sebuah style menggunakan CSS.

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <title>Div Element</title>
5. </head>
6. <body>
7. <div>
8. <p>
9. Paragraf ini berada di dalam elemen div, tetapi akan ditampilkan sama seperti paragraf
10. biasanya.
11. </p>
12. </div>
13. </body>
14. </html>

**Span**

Elemen <span> memberikan manfaat yang sama seperti <div>**,**bedanya elemen ini digunakan sebagai *phrase elements* dan tidak terdapat *line breaks* ketika menggunakannya. Sederhananya, <span> merupakan sebuah <div> yang digunakan dalam sebuah baris teks yang dapat diwadahi oleh *paragraf*, *list***,** *heading*, atau lainnya.

1. <ul>
2. <li>Agil: <span class="phone">08123xxx</span></li>
3. <li>Widy: <span class="phone">08222xxx</span></li>
4. <li>Gilang: <span class="phone">08333xxx</span></li>
5. </ul>

**Tabel**

Elemen <table> pada HTML merepresentasikan data tabular, yaitu informasi yang disajikan dalam sebuah tabel. Tabel sendiri disajikan dalam dua dimensi terdiri dari *baris* dan *kolom* (*cell*) yang berisikan sebuah data. Berikut adalah contoh data sepak bola yang disajikan dalam bentuk tabel.

[A screenshot of a spreadsheet

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32135)

**Struktur Dasar Tabel**

Tabel pada HTML disusun dari tiga buah elemen, yaitu <table>, <tr> dan <td> atau <th>. Elemen <table> digunakan untuk menandakan dimulainya dan diakhirinya sebuah konten tabel dan juga sebagai wadah untuk tabel itu sendiri. Kemudian elemen <tr> digunakan untuk membuat sebuah baris baru yang di dalamnya terdapat elemen <td> atau <th>sehingga menghasilkan sebuah sel.  
  
[A table with blue and pink squares

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32135)

**Spanning Cell**

Dalam aplikasi seperti Microsoft Word, hal ini biasa kita kenal sebagai *merging cell* atau menggabungkan sebuah sel. Ini memang menjadi sebuah fitur dasar dalam membuat sebuah tabel sehingga pada HTML pun kita dapat melakukan hal tersebut.

Pada HTML hal ini lebih dikenal sebagai *spanning cell*, yang artinya menjangkau atau merentangkan sebuah ukuran sel lebih dari ukuran yang biasanya.

* [**Colspan**](https://www.dicoding.com/academies/123/tutorials/32135#tab8-code1)
* [Rowspan](https://www.dicoding.com/academies/123/tutorials/32135#tab8-code2)

1. <table>
2. <tr>
3. <th>18:00</th>
4. <th>19:00</th>
5. <th>20:00</th>
6. </tr>
7. <tr>
8. <td colspan="2">Avenger Infinity Wars</td>
9. <td>It Chapter 2</td>
10. </tr>
11. <tr>
12. <td>One Piece: Stampede</td>
13. <td>Weathering With You</td>
14. <td>Gundala</td>
15. </tr>
16. <tr>
17. <td>Gundala</td>
18. <td colspan="2">Avenger Infinity Wars</td>
19. </tr>
20. </table>

**Input User**

Hampir seluruh website memiliki elemen input. Data yang dimasukkan akan diproses untuk kebutuhan operasional website dalam menunjang pekerjaan manusia. Ada banyak sekali macam-macam input di dunia ini. Namun, jangan khawatir! HTML menyediakan beragam input yang dapat dimanfaatkan. Berikut daftarnya.

**Input Element**

Elemen <input> merupakan elemen yang sangat sering dipakai untuk mendapatkan data dari user. Mengapa hal tersebut terjadi? Hal ini karena elemen input memiliki banyak sekali tipe-tipenya, mulai dari teks, password, email, search, file, dsb. Tidak hanya itu, dari sekian tipe input, masing-masingnya juga didukung oleh atribut khusus sehingga pembuatan formulir semakin *powerful*.

Berikut adalah contoh penerapan dari input element.

1. <div>
2. Email:
3. <input type="email" />
4. </div>
5. <div>
6. Password:
7. <input type="password" />
8. </div>

**Textarea Element**

HTML memiliki elemen khusus yang memungkinkan user menuliskan teks dalam banyak baris. Kenalilah elemen **<textarea>**! Elemen ini berbeda dengan elemen input sebelumnya. Selain nama elemen yang menjadi pembeda, elemen textarea memiliki tag penutup agar dapat berfungsi dengan baik.

Berikut contoh penerapannya.

1. <textarea>
2. Belajar
3. Dasar
4. Pemrograman
5. Web
6. </textarea>

**Label Element**

Pembuatan elemen input sudah umum jika dijajarkan dengan elemen label. Ada banyak sekali keuntungan jika memberikan keterangan pada masing-masing elemen input. Beberapa keuntungan penerapan label untuk elemen input sebagai berikut.

* Elemen input yang berasosiasi dengan elemen label akan memberikan kemampuan bagi *screen reader* untuk menjelaskan fungsi dari elemen input tersebut.
* Memberikan kemampuan bagi browser untuk mengalihkan langsung pada elemen input saat elemen label yang berasosiasi dengannya ditekan atau klik.

Perhatikan contoh kode berikut.

1. <div>
2. <label for="email">Email</label>
3. <br>
4. <input type="email" id="email" />
5. </div>
7. <div>
8. <label for="password">Password</label>
9. <br>
10. <input type="password" id="password" />
11. </div>

**Atribut pada Elemen Input**

Selain banyaknya variasi elemen input, ada banyak atribut yang tersedia untuk memaksimalkan pembuatan formulir. Ada atribut yang bekerja untuk semua tipe input dan ada atribut yang hanya dikhususkan bagi satu tipe.

Berikut adalah contoh penerapan atribut placeholder dan required.

* [**HTML**](https://www.dicoding.com/academies/123/tutorials/32135#tab9-code1)
* [Hasil](https://www.dicoding.com/academies/123/tutorials/32135#tab9-code2)

1. <div>
2. <label for="email">Email</label>
3. <br />
4. <input type="email" id="email" placeholder="example@mail.com" required />
5. </div>
6. <br />
7. <div>
8. <label for="password">Password</label>
9. <br />
10. <input type="password" id="password" placeholder="\*\*\*\*\*\*\*\*" required />
11. </div>

**Mengirim Data Formulir**

Berbicara mengenai mengirim data dari formulir ke server, kita membutuhkan satu elemen esensial dalam membangun formulir. Sebelumnya, kita hanya belajar tentang variasi elemen input untuk mendapatkan data yang sesuai. Ada satu elemen yang berfungsi sebagai wrapper (pembungkus) dari keseluruhan kolom input atau formulir. Elemen yang dimaksud adalah <form>.

Jika diterapkan, berikut adalah contoh sederhana penerapan elemen form.

1. <form>
2. <div>
3. <label for="email">Email</label>
4. <br />
5. <input type="email" id="email" />
6. </div>
8. <div>
9. <label for="password">Password</label>
10. <br />
11. <input type="password" id="password" />
12. </div>
14. <button type="submit">Submit</button>
15. </form>

Dalam mengirimkan data ke server, kita wajib menerapkan atribut name pada seluruh kolom input dalam formulir. Selain itu, ada dua hal yang perlu dilakukan untuk mengirim data ke server, yaitu menetapkan atribut action dan method pada elemen **<form>**.

**Special Character**

Ada beberapa karakter spesial seperti *copyright symbol* (©) yang tidak termasuk ke dalam standar kelompok *ASCII characters*. *ASCII characters* hanya menyediakan karakter seperti huruf, nomor, dan beberapa simbol dasar lainnya.

HTML memerlukan sebuah *“escaped” character* untuk menampilkan karakter khusus. Ada dua cara untuk melakukannya, yakni menetapkan nilai numerik (*numeric entity*) atau menggunakan nama singkatan yang sudah ditetapkan untuk masing-masing karakternya (*named entity*). Semua referensi karakter dimulai dengan “&” dan diakhiri dengan “;”.

Perhatikan contoh kode berikut.

1. <p>Belajar Dasar Pemrograman Web &copy; 2023, Dicoding</p>

**Rangkuman Pengenalan CSS**

Kita telah belajar banyak hal dalam modul Pengenalan CSS. Kami siapkan rangkuman materi terhadap hal yang telah Anda pelajari selama ini. Harapannya, Anda dapat mengingat kembali setiap materi yang disampaikan.

**Pengantar Pengenalan CSS**

Website tampak begitu membosankan dan “mengerikan” tanpa adanya CSS. CSS atau Cascading Style Sheet merupakan standar dari W3C untuk mengatur visualisasi berkas yang ditulis pada HTML. CSS hanya sebuah *declarative language* yang digunakan untuk mendeklarasikan suatu nilai yang nantinya digunakan untuk mengatur tampilan sebuah elemen HTML pada browser.

[A blue and white logo

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32140)

**Keuntungan dan Cara CSS Bekerja**

Dengan menerapkan CSS, tampilan website kita akan lebih menarik. Berikut adalah beberapa keuntungan yang didapatkan ketika kita menerapkan CSS.

* **Dapat mengontrol dan menerapkan layout secara presisi.**  
  Dengan menggunakan CSS kita bisa membuat sebuah website seperti dokumen cetak dengan desain yang menarik dan presisi.
* **Menghindari pekerjaan yang berulang-ulang dalam menerapkan styling.**  
  Kita dapat menetapkan styling pada beberapa berkas HTML hanya dengan menggunakan satu berkas CSS.
* **Didukung banyak browser.**  
  Seluruh browser saat ini minimal sudah mendukung CSS versi 2. Untuk browser yang populer, seperti Chrome dan Firefox, sudah mendukung CSS versi 3.

**Menulis Aturan Styling**

Pada penggunaan CSS, ada dua bagian dalam sebuah rule. Pertama adalah identitas elemen atau elemen yang akan menerapkan rule(singkatnya kita akan sebut selector). Kedua adalah deklarasi atau instruksi yang akan diterapkan pada sebuah selector.

[A different type of fonts

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32140)

**Melampirkan Styling pada Dokumen HTML**

Berikut adalah cara melampirkan styling pada dokumen HTML.

**External Style Sheet**

External Style Sheet adalah berkas terpisah yang di dalamnya hanya ada CSS rules. Berkas ini harus berekstensi **.css**, dan akan dihubungkan dengan dokumen HTML. Cara ini merupakan yang paling *powerful* dalam menerapkan styling. Dengan cara ini, satu berkas styling (.css) dapat digunakan oleh banyak berkas HTML.

1. <head>
2. <meta charset="UTF-8" />
3. <title>Judul Dokumen</title>
5. <!-- Impor berkas CSS Anda -->
6. **<link rel="stylesheet" href="style.css">**
7. </head>

**Embedded Style Sheet**

Embedded Style Sheet adalah kumpulan rulesyang dituliskan dalam berkas HTML dengan menggunakan elemen <style>. Penulisan rules harus dituliskan dalam elemen <style> dan biasanya ditempatkan dalam <head> pada berkas HTML.

1. <head>
2. <meta charset="UTF-8" />
3. <title>Judul Dokumen</title>
5. <style>
6. h1 {
7. color: green;
8. }
9. </style>
10. </head>

**Inline Style**

Inline Style adalah styling yang diterapkan pada elemen HTML dengan menggunakan atribut style. Contohnya seperti berikut.

1. <nama-elemen style="color: green">Konten dari Elemen HTML</nama-elemen>

Untuk menambahkan styling properties lainnya (multiple properties), kita tuliskan dengan menggunakan *semicolon* (**;**) sebagai pemisah antar styling properties-nya.

**CSS Conception**

Berikut adalah rangkuman penjelasan materi dalam materi ini.

**Inheritance**

Styling HTML bersifat inheritance yang berarti dapat mewarisi properti style “tertentu” dari suatu elemen ke elemen-elemen di dalamnya (child-elements).

Contohnya, CSS rulesyang ditetapkan untuk elemen <body> akan diterapkan pada seluruh child-elements secara otomatis.

1. body {
2. font-family: sans-serif;
3. }

Hasilnya berikut.

[A screenshot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32140)

**Group Selector**

Jika beberapa selector yang berbeda memiliki penerapan properti-properti yang sama, kita dapat menggabungkannya menggunakan group selector. Hal ini dapat meminimalkan penulisan kode yang berulang.

Contohnya berikut. Gunakan tanda koma (,) untuk memisahkan tiap selector-nya.

1. h2,
2. h3 {
3. color: #00A2C6;
4. }

**Rule Order**

Cascading artinya “mengalir”. Demikian halnya dengan alur kerja CSS dalam membaca kode, mengalir dari atas ke bawah. Oleh karena itu, kita harus memperhatikan urutan dalam penulisan rules, terutama saat terjadi sebuah konflik.

1. p {
2. color: red;
3. }
5. p {
6. color: blue;
7. }

**Rangkuman Pendalaman CSS**

Kita telah belajar banyak hal dalam modul Pendalaman CSS. Kami siapkan rangkuman materi terhadap hal yang telah Anda pelajari selama ini. Harapannya, Anda dapat mengingat kembali setiap materi yang disampaikan.

**Selector Dasar**

Berikut adalah berbagai selector dasar yang telah dipelajari.

**Type Selector**

Type Selector menggunakan nama elemen sebagai target untuk diterapkannya rule. Dalam kata lain, ketika menggunakan selector ini tentu rule akan diterapkan pada seluruh elemen target yang ada pada dokumen HTML.

**Class Selector**

Class selector menetapkan target elemen berdasarkan nilai dari atribut **class**yang diterapkan pada elemennya. Untuk penulisan selector-nya diawali dengan tanda titik (.) dan dilanjutkan dengan nama class-nya.

**ID Selector**

Class selector menetapkan target elemen berdasarkan nilai dari atribut **id**yang diterapkan pada elemennya. Untuk menetapkan selector dengan menggunakan id, kita gunakan tanda *octothorpe* (#) atau lebih familiar disebut dengan *hash*.

**Attribute Selector**

Attribute selector merupakan cara menetapkan target elemen berdasarkan sebuah atribut yang digunakan atau bahkan bisa lebih spesifik dengan nilainya.

**Universal Selector**

Universal selector digunakan untuk diterapkan pada seluruh elemen. Namun, selector ini juga bisa secara spesifik menargetkan sebuah elemen dengan menggabungkannya bersama selector yang lain.

**Combinators**

Berikut adalah berbagai combinator yang telah dipelajari.

**Adjacent Sibling Selector (+)**

Adjacent Sibling Selector menggabungkan dua buah basic selector dengan menggunakan tanda + di antara keduanya. Contohnya berikut.

1. img + p {
2. color: green;
3. }

Adjacent Sibling Selector terdiri dari dua buah target elemen, tetapi hanya elemen kedua yang menerapkan rule selama elemen tersebut dituliskan langsung setelah elemen pertama pada berkas HTML. Selain itu, kedua elemen tersebut harus berada dalam induk elemen yang sama. Pada contoh di atas, rule akan diterapkan pada elemen paragraf yang berada tepat setelah elemen gambar.

**General Sibling Selector (~)**

Mirip seperti Adjacent Sibling Selector, tetapi rules akan diterapkan pada seluruh elemen kedua yang berada setelah elemen pertama selama masih memiliki induk yang sama, walaupun posisi dari elemen kedua tidak berada tepat setelahnya. General Sibling Selector menggunakan simbol tilda (~) untuk menetapkan elemennya. Biasanya karakter ini berada di tombol kibor sebelah kiri tombol angka “1”.

**Child Selector (>)**

Child Selector menggabungkan dua buah basic selector dengan menggunakan tanda *greater than* (>) di antara basic selectornya. Contohnya sebagai berikut.

1. div > p {
2. background-color: yellow;
3. }

Sebagaimana contoh di atas, CSS rule akan diterapkan pada seluruh elemen paragraf yang berada dalam elemen <div> **secara langsung**. Dalam arti lain, elemen paragraf merupakan ***child*** dari elemen <div>, bukan hanya sebuah turunannya.

**Descendant Selector (space)**

Descendant Selector mirip seperti child selector, tetapi hierarkinya lebih luas karena rule akan diterapkan pada seluruh elemen yang menjadi turunannya walaupun secara tidak langsung. Basic selector pertama yang dituliskan pada selector ini menjadi induknya dan basic selector yang kedua akan menerapkan rule. Selector ini menggunakan spasi dalam menggabungkan dua basic selector. Contohnya berikut.

1. div p {
2. background-color: yellow;
3. }

Dalam contoh di atas, CSS rule akan diterapkan pada seluruh elemen paragraf yang merupakan turunan dari elemen <div>.

**Pseudo Selector**

Berikut adalah macam-macam pseudo selector yang telah dipelajari.

**Pseudo-class Selector**

Pseudo-class merupakan sebuah class “semu” yang sebenarnya ada pada tiap elemen HTML. Pada contoh sebelumnya, kita sudah mengetahui salah satu pseudo-class selector. Dengan menggunakan selector ini kita dapat memilih elemen berdasarkan class yang tidak tampak pada dokumen. Kita bisa menetapkan rule hanya ketika sebuah tautan telah dikunjungi (:visited) atau ketika sebuah elemen diarahkan dengan kursor (:hover).

Untuk menggunakan pseudo-class, kita gunakan tanda titik dua (:) pada basic selector dan diikuti dengan pseudo-class-nya.

**Pseudo-elemen Selector**

Sama seperti pseudo-class, pseudo-element merupakan sebuah elemen “semu” yang sebenarnya ada, tetapi tidak tampak secara tertulis pada berkas HTML. Selector ini biasa digunakan ketika kita ingin menambahkan konten tepat sebelum dan setelah sebuah elemen paragraf. *Yup*, kita tidak perlu menuliskan elemen/konten baru pada berkas HTML untuk menambahkan konten. Namun, kita dapat memanfaatkan pseudo-element untuk melakukannya. Pseudo-element yang dimaksud adalah ::before dan ::after.

**Font Styling**

Ketika kita membuat sebuah dokumen teks, termasuk dokumen cetak, langkah awal yang kita lakukan biasanya menentukan jenis font yang akan digunakan; pada pengembangan website pun demikian. Dalam CSS, font ditentukan menggunakan beberapa paket properti font yang dapat mengatur tipe font, ukuran, ketebalan, dan *style*. Berikut daftarnya.

* 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 caps*(huruf kapital kecil).
* font: sebagai *shorthand* dari properti font yang ada.

**Text Styling**

Setelah mempelajari cara memanipulasi font, kita akan mempelajari cara melakukan formatting pada keseluruhan teks yang ada pada baris paragraf, seperti menetapkan indentasi, jarak antar baris, kata dan huruf, dan sebagainya. Berikut daftar dan penjelasannya.

* Line-height: mengatur jarak minimal dari garis dasar ke garis dasar dalam menampilkannya teks pada halaman.
* Text-indent: memberikan jarak/lekuk di awal paragraf.
* Text-align: mengatur perataan pada teks.
* Text-decoration: menetapkan dekorasi seperti underline (garis bawah), strikethrough (teks dicoret), dan overline (garis atas).
* Text-transform: mengubah kapitalisasi tanpa harus menuliskan kembali teksnya.
* Word and Letter Spacing: mengatur spasi atau jarak pada kata dan huruf.
* Text-shadow: membuat bayangan pada teks.

**Menetapkan Nilai Warna**

Pada CSS, kita dapat menggunakan numeric value dan predefined color name untuk menetapkan nilai warna.

**Numeric Value**

Penetapan warna menggunakan *numeric value* adalah cara yang paling banyak digunakan karena dengan menggunakannya, kita dapat menentukan warna yang diinginkan dengan tepat. Terlebih dengan bantuan *color picker*, kita tidak perlu repot-repot memahami dan mencari nilai kode warna secara manual.

Berikut adalah 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%);

**Predefined Color Name**

Selain menggunakan format angka, menentukan warna juga dapat 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 dapat kita hafal.

**Warna untuk Teks dan Latar Belakang**

Kita sudah belajar dua cara menetapkan warna. Nah, kita perlu mengetahui obyek yang akan diwarnai untuk mengimplementasikan ilmu tersebut, bukan? Ada dua hal yang biasa dilakukan, yaitu teks dan background.

* **Text Color**  
  Kita dapat dengan mudahnya mengganti warna teks menggunakan properti color. Nilai dari properti ini dapat berupa *predefined color name* atau *numeric value*dengan menggunakan RGB, RGBA, Hex, HSL, ataupun HSLA.
  1. p { color: gray; }
  2. p { color: #666666; }
  3. p { color: #666; }
  4. p { color: rgb(102,102,102); }
* **Background Color**  
  CSS memperlakukan setiap elemen HTML seperti berada pada sebuah kotak dan properti **background-color**dapat mengatur warna untuk background dari kotak tersebut.
  1. .box {
  2. background-color: salmon;
  3. }

**Opacity**

Opacity adalah seberapa besar tingkat terlihat suatu objek. Semakin besar tingkat transparansi suatu objek, semakin tak terlihat objek tersebut. Namun, makin besar tingkat opacity suatu objek maka makin terlihat objek tersebut (solid).

1. p {
2. background-color: red;
3. }
5. p.opacity-50 {
6. opacity: 50%;
7. }
9. p.opacity-100 {
10. opacity: 100%;
11. }

Hasilnya berikut.

[A screenshot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32145?from=6154)

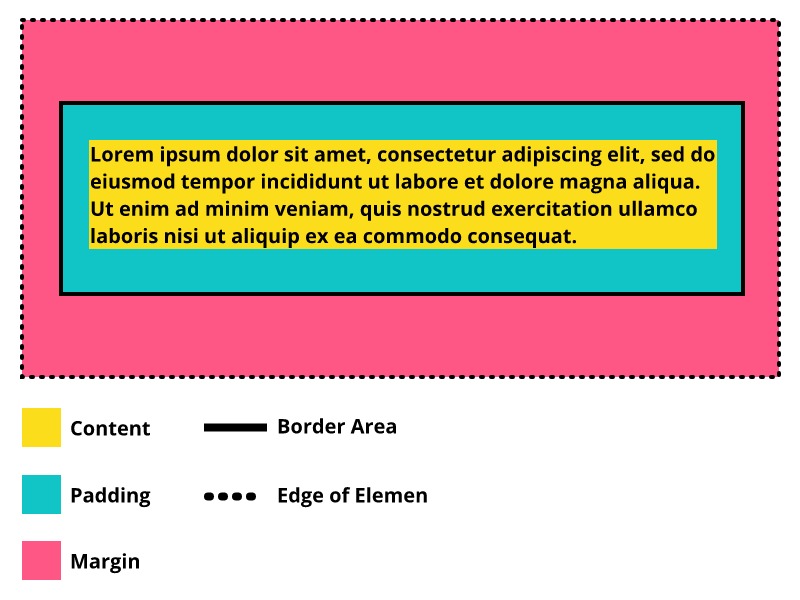
**Box Model**

Setiap elemen yang dibuat pada HTML akan menciptakan sebuah kotak untuk menampung kontennya. Layaknya bentuk kotak pada umumnya, ada beberapa nilai atau komponen padanya.

* Lebar dan tinggi pada kotak (konten).
* Ruang kosong antara konten dengan border (padding).
* Garis tepi (border).
* Jarak dari elemen lain (margin).

Pada CSS, kita dapat mengatur nilai-nilai tersebut. Inilah yang disebut dengan box model.

**Apa itu Box Element**

[](https://www.dicoding.com/academies/123/tutorials/32145?from=6154)

Sebagaimana yang kita lihat pada gambar di atas, setiap elemen pada HTML, baik *block-level* maupun inline-level, akan menghasilkan kotak elemen.

Berikut adalah penjelasan dari gambar di atas.

* Content: konten dari elemen itu sendiri.
* Padding: area yang menjadi jarak antara border elemen dengan konten yang ditampilkan.
* Border: garis yang mencakup konten beserta padding.
* Margin: area jarak di luar border.
* Edge of Element: batas dari suatu elemen.

**Box Model**

Berikut adalah rangkuman materi yang telah dipelajari mengenai box model.

**Dimension**

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

1. .box {
2. height: 300px;
3. width: 300px;
4. }

**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 ditetapkan agar konten selalu ditampilkan secara proporsional. Untuk melakukannya, kita manfaatkan properti **min-width** dan **max-width**.

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

1. .content {
2. max-width: 800px;
3. height: 400px;
4. }

**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.

[A screenshot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32145?from=6154)

**Box-sizing**

Sebelum CSS3, ukuran lebar dan panjang elemen mengacu pada konten elemen (content-box). Itu berarti ukuran elemen seluruhnya merupakan nilai panjang (width) dan lebar (height) yang kita spesifikasikan ditambah dengan nilai padding dan border yang diterapkan pada 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.

1. .content {
2. box-sizing: content-box;
3. }
5. .box {
6. box-sizing: border-box;
7. }

**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.

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

**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.

1. p.example {
2. padding: 10px;
3. }

**Margin**

Margin merupakan ruang atau jarak pada sebuah elemen. Namun, jarak tersebut terletak diluar dari konten dan border element. Margin digunakan untuk menjaga elemen agar tidak bertabrakan satu sama lain atau dari tepi jendela browser.

1. margin: 10px 15px 20px 25px;

**Display Roles**

Standarnya, setiap elemen HTML memiliki dua tipe, yaitu block dan inline.

|  | **Keterangan** |
| --- | --- |
| **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, begitu pun sebaliknya. Properti ini juga dapat digunakan untuk menyembunyikan elemen yang ditampilkan.

**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.

1. .box {
2. width: 200px;
3. height: 200px;
5. background-color: greenyellow;
7. **box-shadow: 6px 6px 5px 10px #666666;**
8. }

**Rounded Corner**

CSS3 memperkenalkan kemampuan untuk membuat *rounded corner* atau sudut bundar pada box dengan menggunakan properti border-radius. Nilai dari properti ini merupakan tingkat lengkungan border dalam piksel.

1. .rounded {
2. width: 300px;
3. padding: 20px;
4. border: 4px solid deeppink;
6. **border-radius: 10px;**
7. }

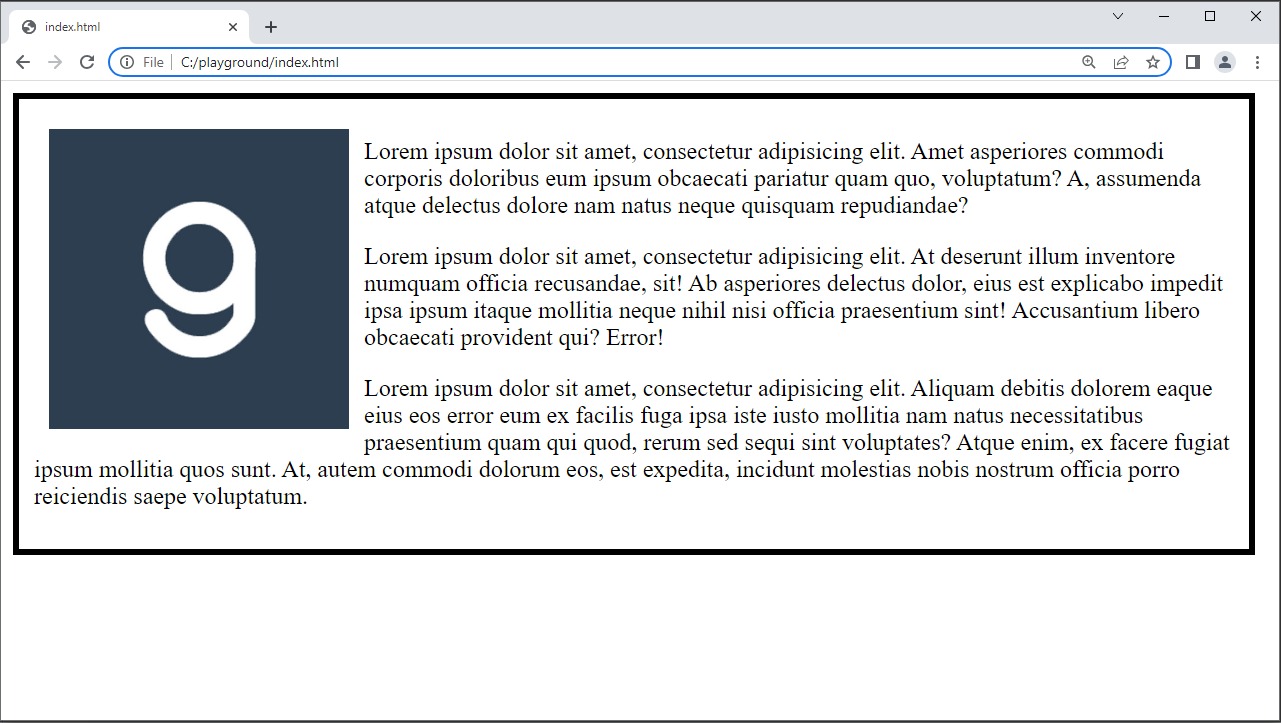
**Positioning**

Penggeseran elemen dengan properti-properti box model akan mempengaruhi keadaan dari elemen di sekitarnya. Untuk melakukan pemindahan elemen secara fleksibel dan bebas, kita perlu mengubah *positioning schema* dengan properti **position**. Berikut adalah nilai dari properti **position** yang ada pada CSS.

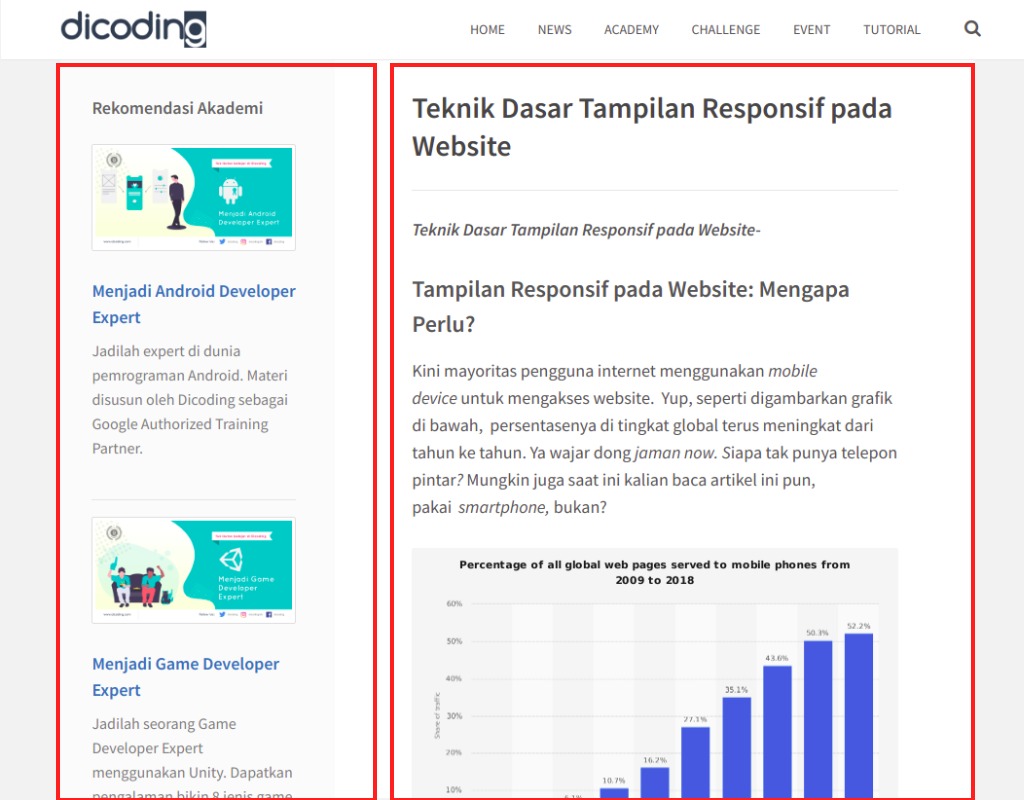
* **Normal Flow/Static Flow**  
  Ia adalah *default behaviour* yang dimiliki elemen, yakni setiap elemen block akan ditampilkan dalam baris baru ketika dibuat. Jadi, setiap elemen block selalu muncul di bawah dari elemen block sebelumnya. Bahkan, jika masih terdapat ruang kosong pada samping elemennya, mereka tidak akan tampak bersebelahan.
* **Relative Positioning**  
  Membuat elemen dapat melakukan perpindahan posisi ke atas, kanan, bawah, ataupun kiri dari posisi semula atau posisi seharusnya elemen tersebut berada. Perpindahan posisi ini tidak akan berpengaruh terhadap posisi elemen di sekitarnya karena ketika menggunakan relative positioning elemen tersebut akan dipindahkan dari *normal flow*.
* **Absolute Positioning**  
  Sama seperti relative, elemen akan dipindahkan keluar dari normal flow sehingga kita dapat memindahkan posisi elemen ke atas, kanan, bawah, ataupun kiri secara leluasa tanpa mengganggu elemen di sekitarnya. Namun, posisinya relatif terhadap jendela browser dan posisinya dapat relatif pada induk elemen selama induk elemen juga berada di luar dari normal flow.
* **Fixed Positioning**  
  Ia merupakan absolute position. Namun, posisinya selalu relatif pada jendela browser, bahkan ketika pengguna melakukan scrolling posisinya akan tetap tampak pada posisinya di layar.

**Floating**

Secara sederhananya, properti float dapat membuat elemen berada pada sebelah kanan atau kiri. Bahkan, jika diterapkan pada elemen inline memungkinkan elemen di sekitarnya mengelilingi elemen tersebut (*wrap*).

[](https://www.dicoding.com/academies/123/tutorials/32145?from=6154)

Selain dalam membuat text wrapping, float juga merupakan salah satu teknik dalam membuat layout website. Teknik ini masih banyak digunakan oleh developer karena sangat mudah dipahami dan tergolong mudah untuk *multiple column* seperti gambar berikut.

[](https://www.dicoding.com/academies/123/tutorials/32145?from=6154)

Hal ini dapat dilakukan dengan mudah menggunakan properti **float** karena nilai dari properti ini adalah **left** dan **right**. Nilai tersebut menunjukkan posisi elemen harus diletakkan.

**Permasalahan Penerapan Float**

Properti float terlihat sangat mudah untuk digunakan, baik dalam *text wrapping* maupun dalam penyusunan layout. Namun, bukan berarti ia tidak memiliki efek samping atau masalah yang ditimbulkan ketika digunakan.

Masalah yang akan ditimbulkan adalah elemen induk akan memiliki tinggi sebesar **0px** jika ia hanya memiliki satu child-element (elemen anak) dengan properti **float**.

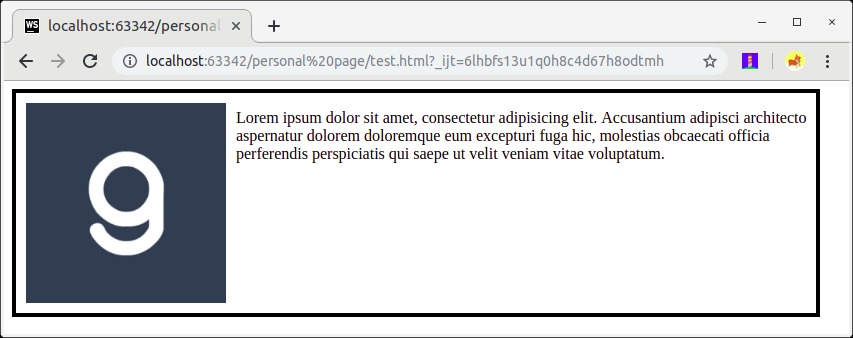
[A screenshot of a computer

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32145?from=6154)

**Clear Property**

Cara yang pertama adalah menggunakan properti **clear**. Properti clear memang dibuat untuk menghilangkan sifat float. Dengan demikian, elemen yang menerapkan properti float akan kembali “dianggap ada”. Namun, untuk menerapkan properti ini kita perlu membuat sebuah elemen kosong (biasanya menggunakan div tanpa konten) yang menerapkan properti clear.

Nilai dari properti **clear** merupakan nilai yang digunakan pada properti **float**. Sebab, properti clear akan menghapus sifat float sesuai dengan nilai yang ditetapkan.

[](https://www.dicoding.com/academies/123/tutorials/32145?from=6154)

**Teknik Overflow**

Cara kedua adalah dengan menetapkan properti **overflow: auto** pada elemen yang menampung elemen float. Sebagian developer lebih memilih menggunakan cara ini karena lebih praktis dan tidak perlu membuat elemen baru ketika menggunakannya.

Untuk menerapkannya sangatlah simpel, hanya tambahkan properti **overflow** dengan nilai **auto** pada container.

**Konfigurasi Meta Tag Viewport untuk Responsibilitas Layout**

Tampilan website kita terlihat baik ketika dibuka menggunakan laptop atau desktop, tetapi belum tentu akan sama pada *mobile device*.

Pada browser Google Chrome dan Mozilla Firefox, ada fitur inspeksi elemen. Fitur ini sangat banyak dan bermanfaat bagi developer web dalam membantu mengembangkan website. Hal yang pasti, pada inspector tersebut, ada fitur yang dapat mensimulasikan halaman website dalam tampilan yang lebih kecil, seperti perangkat mobile atau tablet.

Tampilan pada simulasi device di browser Google Chrome akan tampak seperti berikut.

[A screenshot of a computer

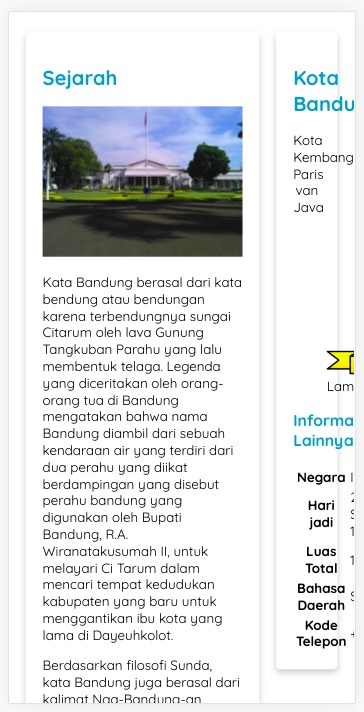
Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32145?from=6154)

Untuk mengatasi masalah di atas, kita perlu menetapkan meta tag viewport untuk mengendalikan dan menentukan bentuk dari viewport browser.

1. <meta name="viewport" content="width=device-width, initial-scale=1">

**Specific Style dengan Media Query**

Walaupun sudah menetapkan meta tag, viewport tampilan pada *mobile device* belum baik. Hal ini karena informasi yang ditampilkan pada elemen <article> terlalu sempit, malahan informasi dalam elemen <aside> sulit terbaca.

[](https://www.dicoding.com/academies/123/tutorials/32145?from=6154)

Untuk mengatasi masalah di atas, CSS menyediakan sebuah fitur untuk menentukan styling hanya pada kondisi browser dan device tertentu yang sesuai dengan aturan yang kita tetapkan. Contohnya, kita memerintahkan ke CSS bahwa “tolong terapkan kode styling ini jika ukuran viewport lebih lebar dari 480 piksel”. Fitur tersebut dinamakan CSS Media Query.

1. @media screen and (max-width: 1000px) {
2. #content, aside {
3. width: 100%;
4. padding: 0;
5. }
6. }

**Rangkuman Layout Responsif dengan Flexbox**

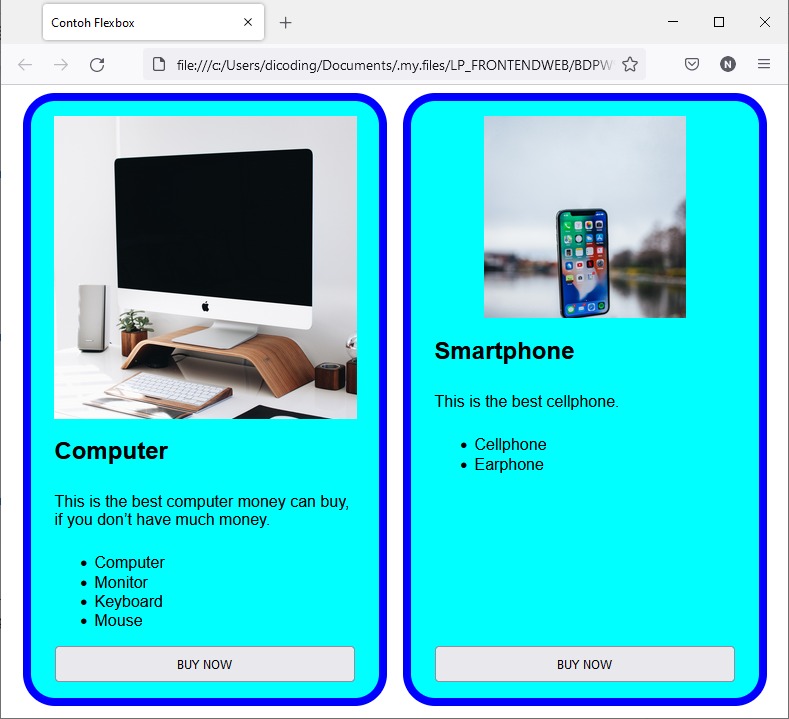
Kita telah belajar banyak hal dalam modul Layout Responsif dengan Flexbox. Kami siapkan rangkuman materi terhadap hal yang telah Anda pelajari selama ini. Harapannya, Anda dapat mengingat kembali setiap materi yang disampaikan.

**Pengantar Flexbox**

**Flexbox**atau **flexible box layout** adalah model layout satu dimensi guna menyusun elemen-elemen (flex items) dalam posisi baris (row) atau baris (column). Hal ini membuat flexbox disebut direction-agnostic.

Hal menarik dari flexbox adalah ia menawarkan penyusunan layout dengan fleksibel karena flex container mengatur dimensi setiap flex-items (child element) sehingga layout yang diinginkan dapat tercapai dengan mudah. Flex container memperluas flex items untuk mengisi ruang kosong atau menciutkannya sehingga menghindari overflow (dimensi child element yang keluar dari parent element-nya).

Dengan hadirnya flexbox, perancangan layout dalam skala yang kecil dapat dilakukan dengan mudah pada halaman web.

[](https://www.dicoding.com/academies/123/tutorials/32150?from=5768)

**Alasan Flexbox Hadir**

Sebelum hadirnya flexbox, satu-satunya cara untuk membuat layout halaman web adalah float dan positioning. Namun, dalam beberapa kasus berikut, penggunaan kedua properti CSS ini terbatas dan menyulitkan.

* Membuat konten berada di tengah secara vertikal dalam parent element-nya.
* Membuat semua child element memenuhi ruang dari parent element-nya secara dinamis dan merata.
* Membentuk child-element dalam layout multiple-column memiliki ukuran height yang sama, meskipun jumlah content di dalamnya tidak sama.

Dari beberapa kasus di atas, layouting pada website akan sulit dijangkau oleh float dan positioning. Di sinilah flexbox hadir untuk menyelesaikan masalah seperti di atas.

**Dasar-Dasar dan Terminologi Flexbox**

[A diagram of a diagram

Description automatically generated](https://www.dicoding.com/academies/123/tutorials/32150?from=5768)

Pada gambar di atas, terlihat bahwa ada dua istilah khusus pada elemen yang menerapkan flexbox, yaitu flex container dan flex item. **Flex container adalah parent elemen dari seluruh flex items**, sedangkan **flex items adalah elemen yang secara langsung merupakan child dari parent element**.

Selain itu, ada beberapa istilah lain dalam flexbox. Berikut penjelasannya.

* Main axis: main axis dari flex container adalah arah sumbu utama yang mana flex items akan dijejerkan atau ditata. Perlu diingat, arah dari main axis tidak selalu horizontal. Hal ini bergantung terhadap value dari properti flex-direction.
* Main-start | main-end: flex items akan ditempatkan dalam flex container yang berawal dari main-start dan main-end.
* Main size: main size merupakan lebar atau tinggi dari flex item yang diposisikan pada sumbu utama (main axis).
* Cross axis: cross axis adalah sumbu yang arahnya tegak lurus dengan main axis. Biasanya ini terjadi ketika flexbox menerapkan value wrap pada properti flex-wrap sehingga susunan layout terlihat dua dimensi.
* Cross-start | cross-end: flex item yang memenuhi ruang kosong flex container secara vertikal, dimulai dari cross-start hingga cross end. Hal ini dapat kita atur menggunakan properti align-items, align-self, dan align-content.
* Cross size: lebar atau tinggi dari sebuah flex item yang ditempatkan pada dimensi cross.

**Properti-Properti pada Flex Container**

Dalam penerapan flexbox, ada properti-properti yang terlibat pada dua hal, yaitu flex container (parent element) dan flex item. Properti-properti pada flex container akan berpengaruh secara langsung kepada flex item dalam menampilkan susunannya. Berikut daftar dan penjelasannya.

* display: mendefinisikan elemen agar disusun sebagai flexible box.
* flex-direction: menentukan arah susunan flex item dijajarkan.
* flex-wrap: mengubah perilaku susunan flex item menjadi dua dimensi (jika dibutuhkan).
* justify-content: mengatur tata letak dari flex item pada main axis.
* align-items: mirip seperti justify-content, tetapi mengatur child element dalam satu baris pada cross axis.
* align-content: melakukan perataan terhadap flex item pada setiap baris dalam cross-axis.
* gap: memberikan jarak atau celah pada flex item.

**Properti-Properti pada Flex Items**

Setelah mempelajari properti-properti pendukung pada flex container, kita akan membahas properti-properti yang digunakan pada flex item. **Ingat!** Properti yang ditujukan pada flex item hanya akan berpengaruh pada flex item yang ditarget, bukan elemen pembungkusnya (flex container).

* order: mengatur urutan susunan flex item.
* flex-grow: melakukan grow (pelebaran ukuran) dari flex item yang ditarget pada main axis.
* flex-shrink: menyusutkan atau menciutkan ukuran child element jika ukurannya tidak mencukupi ruang container.
* flex-basis: memberikan ukuran default sebelum sisa ruang container didistribusikan kepada flex item.
* align-self: mengatur posisi child element secara cross-axis.