Berikut catatan penting yang sudah kita pelajari sejauh ini:

* *Website* : Halaman yang menampilkan informasi kepada pengguna berupa teks atau gambar. Website dapat diakses melalui internet dengan menggunakan *browser*.
* *Browser* : Sebuah perangkat lunak yang dapat menerjemahkan berkas *HTML*, *CSS*, dan *JavaScript* yang didapat dari *web server*untuk ditampilkan dalam bentuk halaman *website*.
* *HTTP Server* : Server berperan pada sebuah *website*sebagai sebuah *software* yang dapat menerima transaksi dari *HyperText Transfer Protocol*.
* *DNS Server* : Server yang dapat mengubah/mengarahkan website melalui sebuah nama domain.
* *Client* : Perangkat yang meminta (*request*) suatu layanan tertentu ke suatu server.
* *HTML* : Salah satu bahasa markup (*markup language)* yang digunakan untuk membuat struktur dan menampilkan konten pada *World Wide Web* (*Website*).
* *CSS* : Bahasa markup yang digunakan untuk mengatur dan mempercantik tampilan pada website.
* *JavaScript* : Bahasa pemrograman yang digunakan untuk membantu *website*menampilkan informasi secara dinamis.
* *Text Editor* : Sebuah perangkat lunak yang digunakan untuk mengelola plain text. Kode HTML, CSS, dan Javascript dituliskan menggunakan perangkat ini.
* *Plain Text* : Teks yang tidak terformat. Format teks ini digunakan dalam penulisan berkas HTML, CSS, dan Javascript.
* *Rich Text* : Teks yang terformat. Format teks ini digunakan jika kita ingin menulis menggunakan Microsoft Word atau *teks editor*berbasis *WYSIWYG (What You See Is What You Get)*.
* *Element* : Sebuah komponen pada HTML yang ditandai dengan tag pembuka dan penutup.

1. <p lang="id" **translate="no"**>Kota metropolitan terbesar di Provinsi Jawa Barat, sekaligus menjadi ibu kota provinsi tersebut.</p>
2. Dengan menambahkan atribut translate dan memberikan nilai **“no”** pada elemen paragraf tersebut, maka konten dari elemen yang dimaksud tidak akan diterjemahkan oleh layanan sistem translate otomatis seperti Google Translate.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| accesskey | Menentukan tombol shortcut untuk mengaktifkan/memfokuskan pada sebuah element. |
| class | Menentukan satu atau lebih classname untuk sebuah elemen. |
| contenteditable | Menentukan konten dari elemen merupakan konten yang dapat diubah atau tidak. |
| data-\* | Digunakan untuk menyimpan sebuah data pribadi khusus ke halaman atau aplikasi. |
| dir | Menentukan arah teks untuk konten pada suatu elemen. |
| draggable | Menentukan apakah suatu elemen dapat di-drag atau tidak. |
| dropzone | Menentukan apakah data yang di-drag adalah data yang disalin, dipindahkan, atau ditautkan saat dijatuhkan. |
| hidden | Menentukan apakah suatu elemen ditampilkan atau tidak pada browser. |
| id | Menetapkan id pada elemen. |
| lang | Menentukan bahasa pada konten elemen. |
| spellcheck | Menentukan apakah elemen harus diperiksa ejaannya dan tata bahasanya atau tidak. |
| style | Menentukan styling secara satu baris untuk suatu elemen. |
| tabindex | Menentukan urutan dari suatu elemen. |
| title | Menentukan informasi tambahan tentang suatu elemen. |
| translate | Menentukan apakah konten elemen harus diterjemahkan atau tidak. |

Pada HTML terdapat tiga tipe list:

* *Unordered lists* : daftar yang ditampilkan tidak memiliki urutan.
* *Ordered lists* : daftar yang ditampilkan secara terurut.
* *Description lists* : daftar yang terbuat dari beberapa istilah diikuti dengan deskripsi dari istilah tersebut.

|  |  |
| --- | --- |
| **Nilai** | **Deskripsi** |
| 1 | Menggunakan angka dalam urutan item (default) |
| a | Menggunakan huruf kecil dalam urutan item |
| A | Menggunakan huruf besar dalam urutan item |
| i | Menggunakan huruf romawi kecil dalam urutan item |
| I | Menggunakan huruf romawi besar dalam urutan item |

Pada ordered list, tipe urutan angkanya dapat kita atur melalui sebuah atribut type. Contohnya, selain nomor urutan angka dapat menggunakan alfabet ataupun angka romawi.

Selain tipe angka pada urutan, kita juga bisa menetapkan nilai awal pada sebuah ordered list dengan menggunakan atribut start. Contohnya, jika kita ingin memulai sebuah list dari angka 7, maka kita tetapkan atribut start dengan nilai 7 pada elemen <ol>.

Normalnya urutan list diawali dengan urutan paling rendah dengan menambahkan atribut reversed pada elemen <ol>

**Teks Terformat**

Sejauh ini, kita sudah mengenal paragraf, heading dan juga list pada HTML. Tapi 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>.

Pada elemen ini kita dapat menggunakan atribut cite untuk menentukan sumber URL

### Preformatted text

Pada sub-modul sebelumnya, kita sudah mengetahui bahwa HTML akan mengabaikan penulisan spasi yang dituliskan secara berulang dan juga line breaks (baris baru). Tetapi pada beberapa tipe konten seperti contoh kode atau puisi hal tersebut sangat berarti. Dengan begitu, terdapat 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. Perhatikan contoh berikut:

### Figure

Elemen ini digunakan untuk merepresentasikan konten tersendiri (self-contained content) seperti ilustrasi, diagram, foto atau bisa juga sebuah baris kode. Banyak hal yang dapat digunakan dalam elemen ini.

Elemen ini digunakan untuk mengelompokkan blok konten yang dapat dipindahkan posisinya dari blok utama sebuah dokumen tanpa mempengaruhi arti dari induk dokumen.

Di dalam elemen figure kita dapat menuliskan elemen <figcaption> sebagai sebuah caption (judul) untuk konten tersebut. Berikut contoh penggunaan figure pada sebuah konten gambar.

**Inline Formatting Text**

Pada materi kali ini, kita akan mengenal beberapa *formatting text* yang digunakan dalam sebuah baris teks (*inline text*). Sebelum menjelaskan elemen inline untuk *formatting text* yang dapat digunakan, sepertinya kita perlu membahas sekilas mengenai *block* dan *inline*.

|  |  |  |
| --- | --- | --- |
| **Atribut** | **Nilai** | **Deskripsi** |
| download | Filename | Menginstruksikan browser untuk mengunduh pada URL yang ditetapkan daripada mengarahkannya. |
| href | URL | Menetapkan target yang akan diarahkan/unduh ketika pengguna menekan hyperlink. |
| hreflang | language\_code | Menetapkan bahasa dari dokumen target. |
| ping | list\_of\_URLs | Menetapkan URL yang akan diberitahu dengan mengirimkan post request ping pada body oleh browser (berjalan di belakang layar) ketika target URL pada hyperlink ditekan. Biasanya atribut ini digunakan untuk pelacakan. |
| referrerpolicy | no-referrer,  no-referrer-when-downgrade,  origin,  origin-when-cross-origin,  unsafe-url | Menetapkan referensi untuk dikirim pada target. |
| rel | alternate,  author,  bookmark,  external,  help,  license,  next,  nofollow,  noreferrer,  noopener,  prev,  search,  tag | Menetapkan hubungan antara halaman yang ditampilkan dengan target. |
| target | \_blank,  \_parent,  \_self,  \_top | Menetapkan lokasi ketika membuka target contohnya pada sebuah tab, window atau pada tab itu sendiri. |
| media | media\_type | Menetapkan tipe media yang digunakan pada target. |

### Emphasized text

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

### Important text

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

### Short quotations

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

### Citation

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

### 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>. Berikut contoh penggunaannya:

### Subscript dan Superscript

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

### 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.

### Line Break

Terkadang kita mungkin perlu menambahkan sebuah baris baru pada sebuah baris teks (termasuk di dalam paragraf), tetapi kita mengetahui bahwa browser akan mengabaikan sebuah penulisan spasi ganda ataupun garis baru, sehingga kita memerlukan sebuah tanda yang dapat digunakan untuk memberitahu browser untuk “Tambahkan garis baru di sini!”.

**Menerapkan Anchor pada Navigasi Halaman Profil**

Sebelumnya, pada halaman profil yang kita buat, sudah ada sebuah list sebagai navigasi. Namun demikian, karena belum ada elemen anchor, maka navigasi tersebut belum dapat kita gunakan. Oleh karena itu, mari kita tambahkan elemen anchor pada tiap item *list*-nya.

Silakan buka kembali berkas **index.html** pada *text editor*. Tambahkan elemen *anchor* di setiap item *list*.

***Semantic HTML* - Mengorganisasikan Halaman Konten**

Seperti yang sudah disebutkan pada sub-modul sebelumnya, sebuah website memiliki hierarki konten yang sama seperti dokumen sehari-hari yang kita baca, misalnya seperti majalah, dan koran. Sehingga, hierarki pada sebuah website merupakan hal yang penting. Namun, tentu saja elemen yang terdapat pada HTML perlu kita kelompokkan menjadi beberapa bagian.

### Header dan Footer

Elemen ini dapat kita gunakan untuk:

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

Dan elemen <footer> digunakan sebagai informasi hak cipta.

i elemen <header> digunakan untuk menampilkan nama situs dan navigasi utama.

Selain itu, elemen <header> dan <footer> dapat digunakan pada sebuah elemen <article> atau <section>. Header biasanya menampung judul dan penulis, footer dapat menampung sebuah link untuk membagikan artikel pada sebuah sosial media.

### Main

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

### Nav

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

Tapi tidak menjamin pada sebuah website hanya ada satu navigasi. Contohnya, sebuah akhir artikel pada blog terdapat tautan navigasi menuju artikel yang dianggap relevan dengan artikel yang telah kita baca. Navigasi tersebut tidak dianggap sebagai navigasi yang penting, sehingga kita tidak perlu menggunakan elemen <nav> untuk menampilkannya.

Pada pembahasan header dan footer, kita sudah melihat contoh penggunaan dari elemen <nav> yang diletakkan pada elemen <header>.

### Articles

Elemen <article> bertindak sebagai container untuk independen konten pada sebuah halaman, artinya konten utuh yang tidak terkait dengan konten lain. Bisa saja konten yang dimaksud berupa artikel blog, komentar, forum post, dan konten lainnya.

### Aside

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

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

### Section

Sebuah elemen yang memiliki kesamaan konten dan memiliki 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.

### Span

Yang kedua elemen <span>, elemen ini 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.

## Table

Tapi sebelum itu, apa sih sebenarnya tabel dalam HTML itu? Pembuatan tabel dalam HTML menggunakan elemen <table>. Elemen <table> ini merepresentasikan data tabular atau informasi yang disajikan dalam sebuah tabel. Tabel sendiri disajikan dalam dua dimensi yang terdiri dari baris dan kolom (cell) yang berisikan sebuah data

### Struktur Dasar Sebuah Tabel

Tabel pada HTML disusun dari tiga buah elemen, yaitu <table>, <tr> sebagai "table row", dan <td> sebagai "table data" atau <th> sebagai "Table head". Elemen <table> digunakan untuk menandakan dimulai 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 cell.

## Spanning Cell

Pada HTML hal ini lebih dikenal sebagai Spanning cell yang artinya menjangkau atau merentangkan sebuah ukuran sel lebih dari ukuran yang biasanya. Dengan spanning cell kita dapat membuat sebuah tabel yang lebih kompleks, tetapi akan membuat markup yang kita tulis menjadi sedikit sulit dibaca.

### Column Spans

Untuk merentangkan sebuah kolom (column spanning) kita bisa menggunakan atribut colspan pada elemen <td> atau <th>. Berikut contoh untuk penggunaan atribut colspan sehingga sebuah header mencangkup dua kolom.

Untuk lebih jelasnya, kita bisa tambahkan atribut border pada elemen <table> agar terdapat garis pada tepi selnya. Value yang diberikan pada atribut ini berupa integer dalam pixel.

#### Row Spans

Untuk merentangkan sebuah baris (row spanning) kita dapat menggunakan atribut rowspan. Sama seperti column spanning, tetapi atribut ini akan merentangkan sebuah sel ke bawah.

|  |  |  |
| --- | --- | --- |
| **Elemen dan Atribut** | | **Description** |
| table | | Menetapkan elemen tabel. |
| td | | Menetapkan sebuah sel dalam baris tabel. |
|  | colspan=”number” | Jumlah kolom yang dicakup oleh sel. |
|  | rowspan=”number” | Jumlah baris yang dicakup oleh sel. |
|  | headers=”nama header” | Mengasosiasikan data sel dengan header. |
| th | | Menetapkan header yang terkait dengan baris atau kolom. |
|  | colspan=”number” | Jumlah kolom dicakup oleh header. |
|  | rowspan=”number” | Jumlah row yang dicakup oleh header. |
|  | headers=”nama header” | Mengasosiasikan header dengan header lain. |
|  | scope=”row|col|rowgroup|colgroup” | Mengasosiasikan header dengan baris, kelompok baris, kolom atau kelompok kolom. |
| tr | | Menetapkan sebuah baris pada tabel. |
| caption | | Memberikan judul pada sebuah tabel. |
| col | | Menetapkan sebuah kolom. |
| colgroup | | Menetapkan sebuah kelompok kolom. |
| tbody | | Mengidentifikasi sebuah body dalam tabel. |
| tfoot | | Mengidentifikasi sebuah footer dalam tabel. |
| thead | | Mengidentifikasi sebuah header dalam tabel. |

## Special Character

Terdapat beberapa karakter spesial seperti copyright symbol © yang tidak termasuk ke dalam standar kelompok ASCII characters, di mana ASCII characters hanya menyediakan karakter seperti huruf, nomor, dan beberapa simbol dasar. Karakter lain seperti lebih dari (>) atau kurang dari (<) walaupun tersedia dalam ASCII character, karakter tersebut tidak dapat digunakan secara langsung sebagai konten pada HTML. Hal tersebut dikarenakan karakter tersebut akan terbaca sebagai sebuah tag.

|  |  |  |  |
| --- | --- | --- | --- |
| **Karakter** | **Deskripsi** | **Named Entity** | **Numeric Entity** |
|  | non-breaking space | &nbsp; | &#160; |
| & | Ampersand | &amp; | &#038; |
| ’ | Apostrophe | &apos; | &#039; |
| < | Kurang dari (less-than) | &lt; | &#060; |
| > | Lebih dari (greater-than) | &gt; | &#062; |
| © | Hak cipta (copyright) | &copy; | &#169; |
| ® | Merek dagang terdaftar (registered trademark) | &reg; | &#174; |
| ™ | Merek dagang (trademark) | &trade; | &#8482; |
| £ | Pound | &pound; | &#163; |
| ¥ | Yen | &yen; | &#165; |
| € | Euro | &euro; | &#8364; |
| – | En-dash | &ndash; | &#8211; |
| — | Em-dash | &mdash; | &#8212; |
| ‘ | Kutip tunggal kiri | &lsquo; | &#8216; |
| ’ | Kutip tunggal kanan | &rsquo; | &#8217; |
| “ | Kutip ganda kiri | &ldquo; | &#8221; |
| ” | Kutip ganda kanan | &rdquo; | &#8221; |
| • | Bullet | &bull; | &#8226; |
| ... | Horizontal ellipsis | &hellip; | &#8230; |

Gunakan !important ketika memang benar-benar dibutuhkan saja. Sebaiknya kita pahami aturan urutan pada CSS dengan baik sehingga meminimalisir penggunaan tanda tersebut.

Berikut catatan yang sudah kita pelajari sejauh ini tentang *styling*:

* *Rule* : Sebuah aturan styling yang harus diterapkan pada elemen HTML. Dalam sebuah *rule*terdapat *selector*dan deklarasi properti *styling*.
* *Selector* : Sebuah bagian dari rule, yang mengidentifikasi target elemen untuk menetapkan sebuah rule.
* *Declaration* : Sebuah bagian dari rule, yang terdiri dari pasangan properti dan nilainya.
* *External Style Sheet* : Berkas terpisah yang di dalamnya hanya terdapat sebuah *rules* yang akan digunakan pada website.
* *Embedded Style Sheet* : Kumpulan rules yang dituliskan dalam berkas HTML dengan menggunakan elemen <style>.
* *Inline Style* : styling yang diterapkan pada elemen HTML dengan menggunakan atribut *style*.

## More about Selector

Ada banyak jenis selector untuk menargetkan aturan ke elemen tertentu dalam dokumen HTML. Pada sub-modul sebelumnya, kita sudah mengetahui salah satu cara dasar dalam menggunakan selector. Sekarang, mari kita bahas lebih detail mengenai macam macam selector yang ada.

### Selector Basic

Pada sub-modul sebelumnya kita sudah mengetahui apa itu selector, dan menggunakannya. Yang kita gunakan tersebut merupakan selector basic. Sebenarnya terdapat beberapa macam dari selector basic, yakni:

* Type Selector
* Class Selector
* ID Selector
* Attribute Selector
* Universal Selector
* **Syntax**
* **Description**
* [attr]
* Menargetkan elemen yang menerapkan atribut attr.
* [attr=value]
* Menargetkan elemen yang menerapkan atribut attr dengan nilai value.
* [attr~=value]
* Menargetkan elemen yang menerapkan atribut attr dengan salah satu nilainya adalah value.
* [attr^=value]
* Menargetkan elemen yang menerapkan atribut attr yang nilainya diawali dengan nilai value.
* [attr$=value]
* Menargetkan elemen yang menerapkan atribut attr yang nilainya diakhiri dengan value.
* [attr\*=value]
* Menargetkan elemen yang menerapkan atribut attr yang nilainya mengadung value.

## Pseudo Selector

Selain beberapa selector yang sudah kita pelajari, CSS masih memiliki dua selector lagi yang dapat kita manfaatkan membantu menyeleksi elemen dalam menerapkan sebuah rule, yakni Pseudo-class dan Pseudo-element.

Sebelum menjelaskan lebih detail tentang kedua selector tersebut, perlu kita ketahui pseudo selector berbeda dari selector yang sudah dibahas sebelumnya, selector ini menargetkan elemen pada bagian yang “tidak terlihat” seperti sifat pada elemen, sehingga untuk menetapkannya, kita tidak bisa menggunakan selector biasa. Salah satu contoh yang paling sering kita terapkan adalah :hover, Pseudo Selector tersebut kita gunakan untuk menetapkan rule ketika cursor diarahkan ke target elemen.

### Pseudo-elemen Selector

Sama seperti pseudo-class, pseudo-elemen merupakan sebuah elemen “semu” yang sebenarnya ada tetapi tidak tampak secara tertulis pada berkas HTML. Selector ini biasa digunakan ketika kita ingin menambahkan sebuah konten tepat sebelum dan setelah sebuah elemen paragraf. Alhasil kita tidak perlu menuliskan struktur elemen tersebut pada berkas HTML. Cukup manfaatkan pseudo-elemen ::before dan ::after kemudian tuliskan konten tersebut cukup pada CSS. Contohnya kita ingin menambahkan tanda kutip sebelum dan sesudah elemen blockquote. Seperti ini cara menuliskannya:

Untuk menggunakan pseudo-elemen kita gunakan dua buah tanda titik dua (::) kemudian diikuti dengan pseudo-elemen nya. Sebenarnya kita bisa menggunakan satu buah tanda tanda titik dua (:) namun kita perlu membedakannya dengan pseudo-class. Karena itulah baiknya ketika menggunakan pseudo-elemen, gunakanlah dua buah titik dua (::).

Pseudo-element tidak hanya *::before* dan *::after*. Dengan pseudo-elemen kita juga dapat menentukan rule pada awal karakter konten elemen.

# Formatting Text (Text and Font Styling)

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

**Font Styling**

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

* font-family : Menetapkan jenis font yang akan diterapkan pada target.
* font-size : Menentukan ukuran pada teks.
* font-weight : Menentukan ketebalan pada teks.
* font-style : Menetapkan styling yang diterapkan pada teks.
* font-variant : Menentukan teks untuk menggunakan gaya *small cap*s (huruf kapital kecil).
* font : Shorthand dari properti font yang ada.
* *serif* : jenis font yang memiliki runcing pada garis akhir karakternya. Times New Roman merupakan salah satu jenis serif font.
* *sans-serif* : jenis font yang tidak meruncing pada garis akhir karakternya. Contohnya “Open Sans”, “Fira Sans” dan lainnya.
* *monospace* : jenis font yang memiliki nilai lebar tiap karakternya sama. Consolas merupakan salah satu jenisnya.
* *cursive*: jenis font yang tampak seperti handwriting atau hasil tulisan tangan.
* *fantasy* : jenis font yang merepresentasikan karakteristik yang menyenangkan.
* *system-ui* : jika menerapkan nilai ini maka font yang diterapkan akan sama seperti font yang digunakan pada sistem operasi kita.
* *math* : jenis font yang digunakan untuk penulisan rumus-rumus matematika.
* *emoji* : jenis font yang digunakan untuk menampilkan emoji.
* *fangsong* : jenis font yang menampilkan gaya penulisan Chinese.

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

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

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

**Absolute unit**

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

### font-weight

Setelah mengenal font families dan font size, selanjutnya ada juga font-weight yang digunakan untuk mengatur ketebalan dari font yang ditampilkan

### font-style

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

### 

### font-variant

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

### Shorthand

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

## Text Styling

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

### Line Height

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

### Text Indent

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

### Text Alignment

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

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

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

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

### Text Transform

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

### Text Transform

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

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

### [Quicksand - Google Fonts](https://fonts.google.com/specimen/Quicksand?query=Quicksand)

### Menetapkan ukuran, ketebalan, dan alignment teks pada Halaman Profil

Setelah menentukan tipe font baru dalam menampilkan teks, selanjutnya kita percantik lagi teksnya dengan mengatur ukuran, ketebalan, dan alignment pada teks. Jika Anda sudah menutup berkas **style.css** pada text editor, silakan buka kembali dan ikuti langkah - langkah berikut.

Langkah pertama kita atur ukuran font menjadi 18px pada teks yang berada di elemen <a> yang merupakan turunan dari elemen <nav>. Silakan tambahkan rule dibawah ini.

# 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.

* *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.

### 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.

1. .content {
2. max-width: 800px;
3. height: 400px;
4. margin: 0 auto;
5. background-color: deeppink;
6. }
8. p {
9. font-size: 1.5em;
10. font-weight: bold;
11. }

### 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:

1. <style>
2. div {
3. height: 200px;
4. width: 200px;
5. background-color: lightgreen;
6. }
7. </style>

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.

## 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.

### 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. |  |

### 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.

1. /\* menggunakan rgb format \*/
2. border-color: rgb(80, 138, 212);
4. /\* menggunakan format hex \*/
5. border-color: #4ee717;
7. /\* menggunakan nama warna \*/
8. border-color: red;

## 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. padding-top: 10px;
14. padding-right: 15px;
15. padding-bottom: 10px;
16. padding-left: 15px;

Atau dengan menggunakan shorthand seperti berikut:

1. padding: 10px 15px 10px 15px;

## 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.

1. margin-top: 10px;
2. margin-right: 15px;
3. margin-bottom: 20px;
4. margin-left: 25px;

## Permasalahan penerapan float

Properti float terlihat sangat mudah untuk digunakan, baik dalam text wrapping maupun dalam penyusunan layout. Tetapi bukan berarti ketika menggunakan properti ini, tiada efek samping atau masalah yang ditimbulkan. Apa itu?

Jika sebuah elemen induk hanya memiliki satu elemen dengan menerapkan properti float, ia akan memiliki nilai tinggi **0px**. Contohnya seperti berikut:

1. <!doctype html>
2. <html lang="en">
3. <head>
4. <style>
5. .container {
6. width: 800px;
7. border: 4px solid black;
8. }
10. img {
11. float: left;
12. margin: 10px;
13. }
14. </style>
15. </head>
16. <body>
17. <div class="container">
18. <img src="https://i.imgur.com/cs2BJzw.jpg" width="200px" alt="dicoding">
19. <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Accusantium adipisci architecto aspernatur dolorem
20. doloremque eum excepturi fuga hic, molestias obcaecati officia perferendis perspiciatis qui saepe ut velit
21. veniam vitae voluptatum.</p>
22. </div>
23. </body>
24. </html>

### Clear Property

Cara yang pertama adalah dengan 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 karena properti clear akan menghapus sifat float sesuai dengan nilai yang ditetapkan. Contohnya seperti ini:

1. <!doctype html>
2. <html lang="en">
3. <head>
4. <style>
5. .container {
6. width: 800px;
7. border: 4px solid black;
8. }
10. img {
11. float: left;
12. margin: 10px;
13. }
14. .clear {
15. clear: left;
16. }

### overflow: auto;

Cara yang 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. Cukup tambahkan properti overflow dengan nilai auto pada container.

1. <!doctype html>
2. <html lang="en">
3. <head>
4. <style>
5. .container {
6. width: 800px;
7. border: 4px solid black;
8. overflow: auto;
9. }
11. img {
12. float: left;
13. margin: 10px;
14. }

### Membuat Sticky Navigation

Ada satu fitur yang mungkin perlu diterapkan lagi pada halaman profil yang kita buat. Pada halaman profil sudah terdapat sebuah navigasi yang berfungsi untuk mengarahkan pengguna langsung pada informasi yang dipilih. Tetapi navigasi tersebut agak sulit diakses ketika pengguna melakukan scrolling pada halaman website. Pasalnya, navigasi tersebut selalu berada di bawah jumbotron website. Akibatnya, untuk mengakses navigasi yang dimaksud, pengguna perlu kembali melakukan scrolling ke atas hingga navigasi kembali terlihat.

Dari permasalahan di atas, sepertinya kita perlu meletakan navigasi yang menempel pada jendela browser, sehingga ketika user melakukan scrolling pada halaman website, navigasi tetap dapat dicangkup oleh pengguna untuk digunakan.

Terdapat satu nilai properti position yang belum dibahas pada pembahasan positioning, yaitu sticky positioning yang merupakan nilai kombinasi dari relative positioning dan fixed positioning.

Dengan menggunakan nilai sticky pada properti position, membuat navigasi dapat melekat pada jendela browser (layaknya fixed position) tetapi posisi awalnya tetap menempati posisi di mana ia seharusnya berada (layaknya relative position). Sehingga tampilan navigasi dapat memiliki sifat seperti ini:

1. nav {
2. background-color: #00a2c6;
3. padding: 5px;
4. position: sticky;
5. top: 0;
6. }

Namun ini masih belum cukup, navigasi bar masih belum tertahan ketika pengguna melakukan scrolling halaman website. Bagaimana agar navigasi dapat menempel pada seluruh halaman? Pada latihan sebelumnya, kita telah membuat elemen <header> sebagai induk dari elemen <nav> dan elemen <div> yang memiliki atribut class dengan nilai jumbotron. Elemen <header> tersebut perlu diterapkan properti display dengan nilai inline. Silakan tuliskan rule berikut pada berkas **style.css**.

1. header {
2. display: inline;
3. }

## Flexible Box Model

Jadi apa sebenarnya itu flexbox atau flexible box model? Pada dasarnya flexbox merupakan mode layout yang mudah dan praktis untuk mengatur elemen dengan menggunakan sebuah container.

Sebelumnya kita sudah menggunakan teknik float dalam penyusunan layout. Float memang cukup mudah dipahami tetapi tak cukup praktis untuk digunakan. Pasalnya, kita memerlukan pengukuran dimensi yang tepat agar elemen berada di posisi yang diinginkan.

Dengan Flexbox kita dapat mengatur ukuran secara otomatis dan mampu beradaptasi dengan ukuran container-nya. Dengan kemampuan flexibelnya, tampilan layout yang disusun menggunakan flexbox menjadi mobile-friendly. Selain itu, flexbox dapat dikombinasikan dengan media query, sehingga ia dapat lebih optimal lagi digunakan pada peranti mobile.

Berikut beberapa konsep dari flexbox yang perlu kita ketahui:

* Dapat mengubah ukuran dimensi elemen dengan menyesuaikan ukuran yang cocok bagi ruang kosong yang ada pada container-nya.
* Flexbox is directional agnostic. ini berbeda dengan konsep **block** model di mana elemen selalu ditampilkan secara vertikal dengan membuat baris baru. Ini berbeda pula dengan konsep **inline**model di mana elemen selalu ditampilkan secara horizontal. Dengan flexbox kita dapat melakukan kedua hal tersebut dengan mudah.
* Dibuat untuk menyusun layout yang mobile friendly.

1. background-color: #11C5C6;
2. border: 2px solid black;
3. padding: 20px;
4. border-radius: 10px;
5. margin: 0 auto;
6. }
7. .box {
8. flex-grow: 1;
10. /\* properti lainnya \*/
11. background-color: #FBDD1C;
12. margin: 5px;
13. border: 2px solid black;
14. border-radius: 10px;
15. }
16. </style>
17. </head>
18. <body>
19. <div class="container">
20. <div class="box"></div>
21. <div class="box"></div>
22. <div class="box"></div>

### Flex Container

Flex container merupakan sebuah elemen yang menampung beberapa flex item. Karena Parent-child merupakan relasi yang wajib ada dalam penerapan flexbox, maka setiap flex item harus merupakan anak / child dari flex container.

Untuk membuat sebuah flex container kita gunakan properti **display** dengan nilai **flex.**Dengan demikian seluruh anak dari container tersebut akan menjadi flex item. Berikut contoh penerapan flexbox untuk membuat tampilan seperti gambar di atas:

### Flex Grow

Bisa kita lihat seluruh box di dalam container dapat menyesuaikan ukurannya sesuai ruang kosong yang ada pada container. Pada box, kita tidak menetapkan nilai dimensi seperti width dan height. Kita cukup menggunakan satu properti yakni flex-grow: 1;.

Properti flex-grow ini digunakan untuk memberitahu berapa banyak ukuran yang harus ditetapkan oleh flex-item. **Nilai dari properti ini bukan nilai dari dimensi asli pada flex item, melainkan nilai yang relatif terhadap ruang kosong pada flex container.**

1. <!doctype html>
2. <html lang="en">
3. <head>
4. <style>
5. .container {
6. display: flex;
8. /\* Properti lainnya \*/
9. width: 800px;
10. height: 250px;
11. background-color: #11C5C6;
12. border: 2px solid black;
13. padding: 20px;
14. border-radius: 10px;
15. margin: 0 auto;
16. }
17. .box {
18. /\* properti lainnya \*/
19. background-color: #FBDD1C;
20. margin: 5px;
21. border: 2px solid black;
22. border-radius: 10px;
23. }
25. .first {
26. flex-grow: 1;
27. }
28. .second {
29. flex-grow: 2;
30. }
32. .third {
33. flex-grow: 1;
34. }
35. </style>
36. </head>
37. <body>
38. <div class="container">
39. <div class="box first"></div>
40. <div class="box second"></div>
41. <div class="box third"></div>
42. </div>

### Flex Direction

Seperti yang sudah kita ketahui sebelumnya, flexbox merupakan directional agnostic, di mana kita dapat mengubah arah munculnya flex-item yang berada di flex container. Secara default deretan flex-item ditampilkan secara **horizontal**, namun kita dapat mengubahnya dengan menetapkan properti **flex-direction** pada flex container-nya.

1. <!doctype html>
2. <html lang="en">
3. <head>
4. <style>
5. .container {
6. display: flex;
7. flex-direction: column;
9. /\* Properti lainnya \*/
10. width: 800px;
11. height: 250px;
12. background-color: #11C5C6;
13. border: 2px solid black;
14. padding: 20px;
15. border-radius: 10px;
16. margin: 0 auto;
17. }
18. .box {
19. /\* properti lainnya \*/
20. background-color: #FBDD1C;
21. margin: 5px;
22. border: 2px solid black;
23. border-radius: 10px;
24. }
26. .first {
27. flex-grow: 3;
28. }
29. .second {
30. flex-grow: 2;
31. }
32. .third {
33. flex-grow: 1;
34. }
35. </style>
36. </head>
37. <body>
38. <div class="container">
39. <div class="box first"></div>
40. <div class="box second"></div>
41. <div class="box third"></div>

Ada empat nilai yang bisa digunakan untuk properti **flex-direction**, antara lain:

* **row**: merupakan nilai default, di mana deretan flex-item pada container ditampilkan secara horizontal.
* **row-reverse**: memiliki sifat yang sama seperti row, namun urutan flex item-nya ditukar.
* **column**: Deretan flex-item pada container ditampilkan secara vertikal.
* **column-reverse**: memiliki sifat yang sama seperti column, namun urutan flex item-nya ditukar.

### Flex Basis

Selain menggunakan flex-grow, untuk menentukan ukuran flex item, kita bisa gunakan properti flex-basis. Properti ini mirip seperti width dalam menentukan dimensi box. Kita bisa menggunakan nilai satuan tetap seperti px, pt, pc, cm dll. Selain itu, kita juga bisa menggunakan satuan persentase (%).

Properti flex-basis ini digunakan ketika kita ingin menetapkan ukuran awal pada sebuah flex-item. Alhasil, kita dapat mengatur ukuran dengan lebih leluasa. flex-basis biasa digunakan ketika kita menerapkan nested flex-container dan terdapat perbedaan jumlah child pada container-nya. Untuk lebih mudah menggambarkannya, perhatikan contoh berikut:

1. <!doctype html>
2. <html lang="en">
3. <head>
4. <style>
5. \* {
6. box-sizing: border-box;
7. }
8. .container {
9. width: 800px;
10. border: 2px solid black;
11. margin: 0 auto;
12. }
13. .flex-column {
14. display: flex;
15. flex-direction: column;
16. }
17. .flex-row {
18. display: flex;
19. flex-direction: row;
20. flex-grow: 1;
21. }
22. .box {
23. flex-grow: 1;
24. background-color: cornflowerblue;
25. border: 2px solid black;
26. padding: 40px;
27. text-align: center;
28. color: white;
29. font-size: 1.5em;
30. }
31. </style>
32. </head>
33. <body>
34. <div class="container">
35. <div class="flex-column">
36. <div class="flex-row">
37. <div class="box">1</div>
38. <div class="box">2</div>
39. <div class="box">3</div>
40. <div class="box">4</div>
41. </div>
42. <div class="flex-row">
43. <div class="box">5</div>
44. <div class="box">6</div>
45. <div class="box">7</div>
46. </div>
47. </div>
48. </div>

Jika kita menggunakan flex-grow, mungkin kita dapat memberikan nilai flex-grow: 2 pada box ke tujuh.

1. <!-- ..... -->
2. <div class="box">4</div>
3. </div>
4. <div class="flex-row">
5. <div class="box">5</div>
6. <div class="box">6</div>
7. **<div class="box double">7</div>**
8. </div>
9. </div>
10. <!-- ..... -->
11. <!-- ..... -->
12. .double {
13. flex-grow: 2;
14. }

apat asumsikan bahwa total ruang kosong pada flex-container adalah 100%. Jika dibagi rata terhadap empat buah flex-item, maka tiap itemnya harus memiliki nilai 25%.

1. <!doctype html>
2. <html lang="en">
3. <head>
4. <style>
5. \* {
6. box-sizing: border-box;
7. }
9. .container {
10. width: 800px;
11. border: 2px solid black;
12. margin: 0 auto;
13. }
15. .flex-column {
16. display: flex;
17. flex-direction: column;
18. }
20. .flex-row {
21. display: flex;
22. flex-direction: row;
23. flex-grow: 1;
24. }
26. .box {
27. flex-basis: 25%;
28. background-color: cornflowerblue;
29. border: 2px solid black;
30. padding: 40px;
31. text-align: center;
32. color: white;
33. font-size: 1.5em;
34. }
35. </style>
36. </head>
37. <body>
38. <div class="container">
39. <div class="flex-column">
40. <div class="flex-row">
41. <div class="box">1</div>
42. <div class="box">2</div>
43. <div class="box">3</div>
44. <div class="box">4</div>
45. </div>
46. <div class="flex-row">
47. <div class="box">5</div>
48. <div class="box">6</div>
49. <div class="box">7</div>
50. </div>
51. </div>
52. </div>
53. </body>
54. </html>

## Membuat Layout Kalkulator

Setelah mengenal dasar penggunaan flexbox, mari sekarang kita terapkan apa yang sudah kita pelajari dengan membuat dasar layout kalkulator.

### Membuat struktur HTML kalkulator

Mari kita membuat sebuah project baru dengan membuat folder baru dan beri nama folder sesuai keinginan atau pada contoh kali ini kita beri nama “WebCalculator”. Lalu buka folder tersebut menggunakan teks editor yang Anda miliki dan buat berkas HTML baru dengan nama “index.html”.

Kemudian setiap elemen .button, di dalamnya kita berikan teks yang digunakan untuk melabelkan tiap-tiap tombolnya. Tuliskan teks angka 0 hingga 9 beserta operator-operator yang akan digunakan pada kalkulator dengan urutan seperti berikut.

1. <div class="flex-container-column">
2. <div class="flex-container-row">
3. <div class="button">7</div>
4. <div class="button">8</div>
5. <div class="button">9</div>
6. <div class="button">+/-</div>
7. </div>
8. <div class="flex-container-row">
9. <div class="button">4</div>
10. <div class="button">5</div>
11. <div class="button">6</div>
12. <div class="button">-</div>
13. </div>
14. <div class="flex-container-row">
15. <div class="button">1</div>
16. <div class="button">2</div>
17. <div class="button">3</div>
18. <div class="button">+</div>
19. </div>
20. <div class="flex-container-row">
21. <div class="button">CE</div>
22. <div class="button">0</div>
23. <div class="button">=</div>
24. <div class="button"></div>
25. </div>
26. </div>

Jika kita perhatikan dari beberapa tombol di atas, ada yang berupa tombol angka, operator dan fungsi. Untuk tombol operator seperti tambah (+) dan kurang (-) berikan class tambahan dengan nilai “operator”.

1. .....
2. <div class="button operator">-</div>
3. .....
4. <div class="button operator">+</div>

Kemudian untuk tombol fungsi seperti negative (+/-), clear (CE), dan equals (=) berikan class tambahan dengan nilai sesuai dengan fungsinya tersebut.

1. .....
2. <div class="button negative">+/-</div>
3. .....
4. <div class="button clear">CE</div>
5. .....
6. <div class="button equals">=</div>

Penamaan class yang spesifik seperti ini dibutuhkan untuk memudahkan kita mengambil nilai elemen nanti ketika menggunakan JavaScript, hal ini juga membantu kita untuk memudahkan penentuan selector untuk styling menggunakan CSS.

Sebuah kalkulator tentu tidak hanya terdiri dari tombol saja bukan? Tiap kalkulator pasti memiliki layar untuk menampilkan apa yang pengguna input dan menampilkan hasil kalkulasinya. Dengan begitu kita membutuhkan elemen lain yang berfungsi sebagai layar. Tambahkan elemen div baru tepat di bawah tag pembuka .flex-container-column dan beri atribut class dengan nilai “display”.

1. <div class="flex-container-column">
2. <div class="display"></div>
3. <div class="flex-container-row">
4. .....

Kemudian, di dalam elemen div.display tersebut tambahkan elemen <h1> sebagai teks yang berfungsi untuk menampilkan angka pada layar tersebut.

1. <div class="display">
2. <h1 id="displayNumber">0</h1>
3. </div>

### Menerapkan Flexbox pada Halaman Kalkulator

Setelah kita selesai membuat struktur HTML, selanjutnya kita akan percantik elemen - elemen tersebut dengan styling.

Sebelum memulai menuliskan styling pada CSS, jangan lupa untuk hubungkan berkas style.css sebagai stylesheet pada index.html. Tuliskan tag <link> berikut ini di dalam elemen <head>:

1. <link rel="stylesheet" href="assets/style.css">

Untuk memulai menuliskan styling, kita buka berkas style.css. Kita mulai dengan membuat seluruh elemen menerapkan nilai *border-box* pada properti *box-sizing*. Tuliskan rule berikut pada berkas style.css.

1. \* {
2. box-sizing: border-box;
3. }

Lalu kita tetapkan *font-family* pada seluruh teks yang ada di dalam *body* dengan menggunakan sans-serif.

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

Selanjutnya kita mulai terapkan *flexbo*x pada .flex-container-column dengan memberikan nilai **flex** pada properti **display**. Jangan lupa tetapkan properti flex-direction: column dan properti pendukung lainnya. Sehingga styling .flex-container-column akan tampak seperti berikut:

1. .flex-container-column {
2. display: flex;
3. flex-direction: column;
5. /\* properti pendukung \*/
6. max-width: 800px;
7. margin: 0 auto;
8. text-align: right;
9. }

Terapkan juga properti display: flex pada .flex-container-row. Namun kita tidak harus menerapkan flex-direction pada elemen ini, karena nilai row sudah menjadi default pada flex-container. Sehingga penulisan styling untuk .flex-container-row cukup dituliskan seperti berikut:

1. .flex-container-row {
2. display: flex;
3. }

Mari kita lanjut atur styling untuk tombol kalkulatornya. Pada tiap .button kita akan beri nilai flex-basis sebesar 25% beserta properti pendukung lainnya sehingga tombol kalkulator akan memenuhi seluruh ruang kosong pada container. Tuliskan styling berikut untuk .button:

1. .button {
2. flex-basis: 25%;
4. /\* properti pendukung \*/
5. font-size: 1.5em;
6. text-align: center;
7. padding: 40px;
8. border: 1px solid black;
9. background: -webkit-linear-gradient(top, #d2d2d2, #ddd);
10. cursor: pointer;
11. }
12. \* {
13. box-sizing: border-box;
14. }
16. body {
17. font-family: sans-serif;
18. }
20. .flex-container-column {
21. display: flex;
22. /\* properti pendukung \*/
23. flex-direction: column;
24. max-width: 800px;
25. margin: 0 auto;
26. text-align: right;
27. }
29. .flex-container-row {
30. display: flex;
31. }
33. .button {
34. flex-basis: 25%;
35. /\* properti pendukung \*/
36. font-size: 1.5em;
37. text-align: center;
38. padding: 40px;
39. border: 1px solid black;
40. background: -webkit-linear-gradient(top, #d2d2d2, #ddd);
41. cursor: pointer;
42. }
44. .double {
45. flex-basis: 50%;
46. }
48. .display {
49. color: white;
50. width: 100%;
51. padding: 10px 20px;
52. background-color: #333333;
53. border: 1px solid black;
54. font-size: 2em;
55. }
57. .operator,
58. .negative,
59. .equals {
60. background: orange;
61. }
63. .card {
64. box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2);
65. border-radius: 5px;
66. padding: 30px;
67. margin-top: 20px;
68. }
70. .button:hover {
71. font-weight: bold;
72. }

Sejauh ini berikut kode yang ada pada index.html dan style.css:

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <title>Web Calculator</title>
5. <link rel="stylesheet" href="assets/style.css">
6. </head>
7. <body>
8. <div class="flex-container-column card">
9. <div class="display">
10. <h1 id="displayNumber">0</h1>
11. </div>
12. <div class="flex-container-row">
13. <div class="button">7</div>
14. <div class="button">8</div>
15. <div class="button">9</div>
16. <div class="button negative">+/-</div>
17. </div>
18. <div class="flex-container-row">
19. <div class="button">4</div>
20. <div class="button">5</div>
21. <div class="button">6</div>
22. <div class="button operator">-</div>
23. </div>
24. <div class="flex-container-row">
25. <div class="button">1</div>
26. <div class="button">2</div>
27. <div class="button">3</div>
28. <div class="button operator">+</div>
29. </div>
30. <div class="flex-container-row">
31. <div class="button clear">CE</div>
32. <div class="button">0</div>
33. <div class="button equals double">=</div>
34. </div>
35. </div>
36. </body>
37. </html>