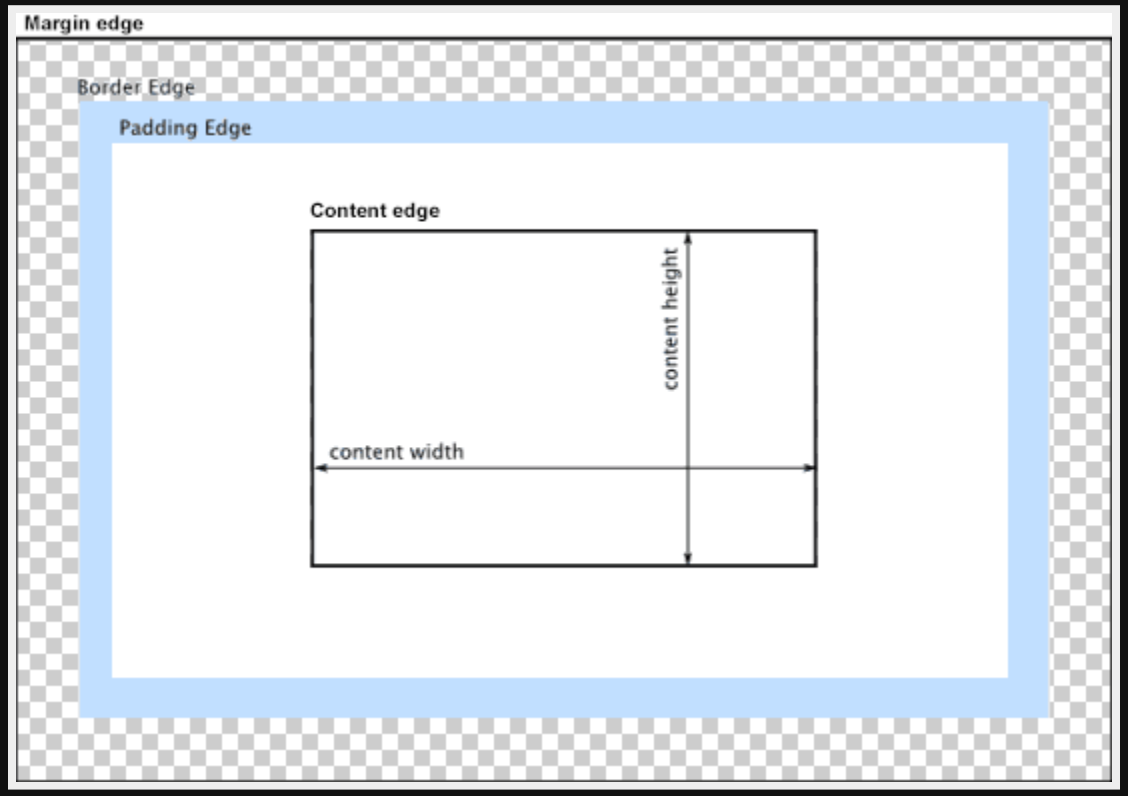
Kursus Coding Froyo Framework

Week 3: CSS Lanjutan

Box Model

- Secara default, ini adalah cara yang diambil oleh browser untuk me-render tampilan sebuah halaman website
- Semua elemen akan dianggap sebagai sebuah elemen box (kotak)
- Model ini mendeskripsikan ukuran tempat yang digunakan oleh sebuah elemen
- Dalam 1 box, ada 4 sisi yang menentukan ukuran box:
  - margin
  - border
  - padding
  - content
- Untuk bisa memahami layout CSS dengan baik, kita perlu memahami model ini

Box Model



.png)

Logika CSS: Content

- Content adalah ukuran sebenarnya milik sebuah elemen
- Meski isinya bukan kotak, ia akan tetap di-render sebagai kotak
- Saat `box-sizing` di-set default, maka untuk mengontrol ukuran content ini, parameter-parameter yang digunakan adalah:
  - `width`
  - `min-width`
  - `max-width`
  - `height`
  - `min-height`
  - `max-height`

Logika CSS: Padding

- Padding adalah ukuran jarak antara content dengan elemen yang menampungnya
- Ia bisa dibayangkan sebagai jarak antara sisi padding dengan sisi content
- Ketika ukuran ini diubah, ia akan nampak mengubah ukuran dari content
- Bila sebuah content memiliki background warna atau gambar, ia akan mengikuti padding juga
- Parameter pengatur padding adalah:
  - `padding-right`
  - `padding-left`
  - `padding-top`
  - `padding-bottom`

Logika CSS: Border

- Border adalah ukuran garis tepi sebuah elemen
- Ia opsional, bisa ada atau tidak
- Jika ada, maka ia akan menambahkan ukuran dari sebuah content
- Parameter pengaturnya adalah:
  - `border-width`
- Untuk menampilkan border, perlu diatur juga `border-style`-nya
  - contoh: `border-style:solid`

Logika CSS: Margin

- Margin adalah jarak antara 1 elemen dengan elemen di dekatnya
- Ia memperluas area dari border ke area kosong yang memisahkan satu elemen dengan tetangganya
- Margin tidak akan memperbesar ukuran yang nampak dari sebuah elemen, tapi jadi mengatur jarak dan posisinya, relatif terhadap elemen tetangganya
- Yang berubah ukurannya, adalah elemen kontainernya
- Parameter pengaturnya adalah:
  - `margin-top`
  - `margin-bottom`
  - `margin-left`
  - `margin-right`

Contoh Kasus:

- **Mendekatkan 2 elemen**
- Kita bisa menggunakan margin untuk melakukan ini
- Margin di sisi mana yang digunakan, bergantung dari elemen mana yang akan diubah CSS-nya

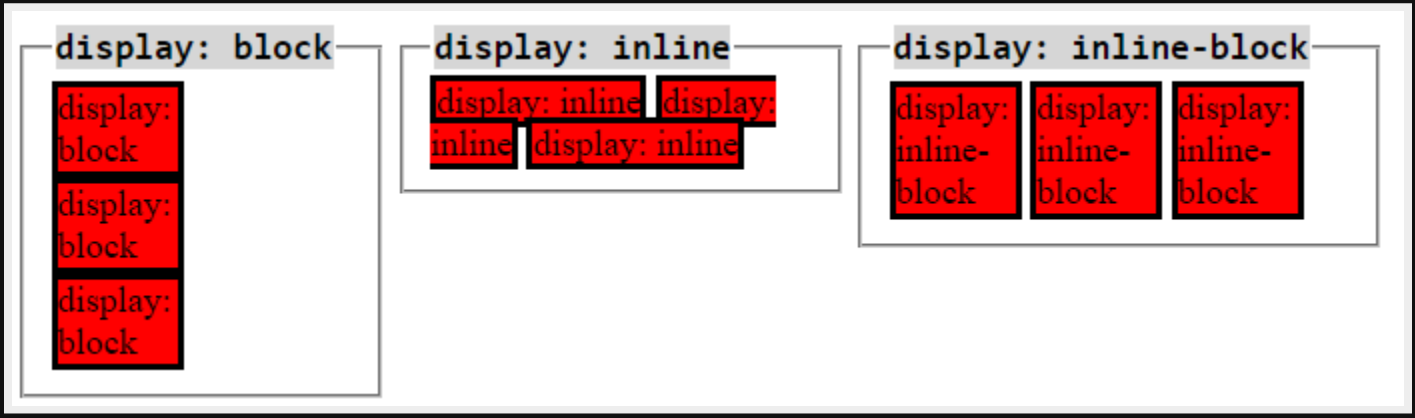
Contoh Kasus:

- **Mengatur posisi elemen**
- Kita bisa menggunakan padding di elemen div kontainer untuk mengatur posisi elemen di dalamnya
- Sebagai tambahan, kita bisa juga mengubah margin milik elemen tersebut

Logika CSS: Display

- `inline`:
  - Semua elemen ditampilkan bersebelahan, dalam 1 baris
  - Tidak memperdulikan margin dan padding di sisi atas dan bawah
  - Tidak bisa mengatur properti `width` dan `height`
- `block`
  - Semua elemen ditampilkan vertikal, dari atas ke bawah, karena memaksa untuk mengganti baris
  - Semua margin dan padding akan diperhitungkan
- `inline-block`
  - Memungkinkan semua elemen untuk bersebelahan
  - Memperdulikan margin dan padding di sisi atas dan bawah
  - Bisa mengatur properti `width` dan `height`

Logika CSS: Display



- Properti ini diatur di dalam elemen yang ingin kita ubah posisinya

Contoh Kasus: Mengatur Elemen agar Bersebelahan

- Apabila kita memiliki 2 `div` yang ingin kita buat bersebelahan, maka keduanya perlu diatur agar properti `display`-nya adalah `inline` atau `inline-block`
- Defaultnya, ini bernilai `block` sehingga posisinya atas-bawah
- Mana properti `display` yang dipilih, bergantung dari konten `div` tersebut

Logika CSS: Float

- `float` mengatur apakah sebuah elemen harus ditarik dari alur render normal (vertikal, dari atas-bawah) dan diposisikan di kiri atau kanan, serta elemen lain diposisikan mengelilinginya
- Menggunakan `float` di sebuah elemen juga otomatis akan mengubah posisi elemen di sebelahnya
- Untuk membuat agar elemen di sebelahnya tidak terkena efek dari `float`, maka kita bisa mengaktifkan `clear` di elemen tersebut, sehingga ia akan ditampilkan di bawah elemen yang memiliki properti `float`

Contoh Kasus: Membuat Layout dengan Float

- Kita bisa menggunakan `float` untuk membuat layout
- Elemen yang diposisikan di paling kiri, bisa diberikan `float: left`
- Elemen selanjutnya bisa diberikan properti serupa
- Elemen yang tidak ingin diletakkan di sebelahnya, bisa diberikan `clear: left`

Logika CSS: Position

- Properti yang mengatur posisi dari sebuah elemen, tanpa menggunakan `margin` ataupun `float`, tapi diatur koordinatnya di halaman
- CSS memiliki 2 tipe `position`, relative dan absolute
- Relative, artinya posisi elemen tersebut akan dihitung dengan mengacu pada posisi seharusnya, apabila properti `position` tidak diubah
- Absolute, artinya posisi elemen tersebut dihitung relatif terhadap elemen kontainernya dan dianggap tidak mengikuti alur elemen di dokumen

Font

- CSS bisa digunakan untuk mengatur font yang digunakan, serta properti yang berkaitan dengannya
- Beberapa hal yang bisa diatur dengan CSS:
- `font-family`: mengatur jenis/nama font
- `font-style`: mengatur tipe font: normal atau italic
- `font-size`: mengatur ukuran font. bila menggunakan em, 1 em = 16 px
- `font-weight`: mengatur apakah font bold atau normal

Contoh Styling Font dengan CSS

```css
p {
  font-family: "Georgia", Serif;
  font-weight: 700;
  font-size: 1.5em;
}
```

- Kita bisa mendefinisikan font yang digunakan dengan memanggil nama font (`"Georgia"`), atau tipenya (`Serif`)
- `font-weight` juga bisa didefinisikan dengan angka atau tipe, seperti: `bold`

Menggunakan Font dari Google Fonts

- Google memiliki layanan penyedia font gratis dan legal yang bisa kita gunakan langsung di project kita
- Kita bisa langsung memilih dan memanggil font tersebut di file CSS kita

Menggunakan Font dari Google Fonts

```
<link href='https://fonts.googleapis.com/css?family=Oswald:400,300,700' rel='stylesheet
<style type="text/css">
p {
  font-family: "Oswald", sans-serif;
  font-weight: 700;
  font-size: 1.5em;
}
</style>
```

- Ketika kita hanya mengimpor font dengan `font-weight` tertentu, hanya tipe tersebut yang bisa digunakan
- Karena di contoh hanya ada style 300, 400 dan 700, maka hanya ketiga tipe font tersebut yang bisa digunakan

Selector Tambahan

- Selain memilih selector dengan menelusuri pohon elemen HTML, atau memilih elemen/id/class tertentu, CSS juga memiliki fitur untuk memilih berdasarkan pola tertentu
- Misal:
- memilih anak pertama dari sebuah elemen,
  - menambahkan konten setelah sebuah elemen
  - memilih elemen yang sedang ditunjuk mouse
  - Kita akan mengenalkan beberapa selector tambahan yang lazim digunakan

Selector Tambahan: `before` dan `after`

- Kedua selector ini digunakan untuk menambahkan sesuatu sebelum atau sesudah sebuah elemen
- Contoh:

```
p:after {
content: " - ini cuma paragraf dalam div";
}
```

Selector Tambahan: `child`

- Dengan ini, kita bisa memilih anak dari sebuah elemen, baik `first-child`, `last-child` atau `nth-child(n)`
- Contoh:

```
p:first-child {
color: #0022dd;
}

<div>
<p>Halo, ini adalah isi pertama dari div ini</p>
<p>Halo, ini adalah isi kedua dari div ini</p>
<p>Halo, ini adalah isi ketiga dari div ini</p>
</div>
```

  - Hanya isi pertama yang warna font-nya berubah

Selector Tambahan: `:active` dan `:hover`

- Dengan ini, kita bisa mengaktifkan properti CSS tertentu untuk sebuah elemen yang sedang ditunjuk mouse (`:hover`) atau yang sedang aktif (`:active`)
- Contoh:

```
a:active {
color: #aa0045;
}

<a href="about.html">About</a>
<a href="contact.html">Contact</a>
```

- Apabila kita sedang berada di halaman home, dan kita mengklik link ke halaman About, maka halaman akan berpindah dan link tersebut menjadi berwarna merah

Transitions

- Transition digunakan untuk menganimasikan perubahan satu atau beberapa properti CSS
- Ini adalah cara dasar untuk melakukan animasi hanya menggunakan CSS
- Dalam transition, kita pilih properti yang ingin dianimasikan dan berapa lama animasi berjalan
- Dianjurkan penggunaannya hanya dalam jumlah kecil dan tidak kompleks, karena cukup berat

Transitions: Contoh

- Contoh berikut akan menganimasikan perubahan warna background dari sebuah div, saat ia ditunjuk oleh mouse (hover)
  - Perubahan berlangsung selama 2 detik

```
#container:hover {
background-color: #dedede;
transition: background-color 2s;
}
```

Responsive Design dengan PureCSS

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

Masukkan Grid System

```
<!--[if lte IE 8]>
<link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/grids-responsive-old-i
<![endif]-->
<!--[if gt IE 8]><!-->
<link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/grids-responsive-min.c
<!--<![endif]-->
```

Grid System di PureCSS

```
<div class="pure_g">
<div class="pure-u-1-3"></div>
<div class="pure-u-1-3"></div>
<div class="pure-u-1-3"></div>
</div>
```

Mengenal Media Queries

```
sm >= 568 px
md >= 768 px
lg >= 1024 px
xl >= 1280 px
```

Flexbox Model (Materi Bonus)

- Flexbox (flexible box) adalah sebuah model layout di CSS
- Ia menyediakan sistem pengaturan elemen di sebuah halaman di mana elemen-elemen ini akan diatur posisinya, mengikuti ukuran layarnya
- Ia menyediakan perbaikan dari model box. Selain itu, ia tidak menggunakan float

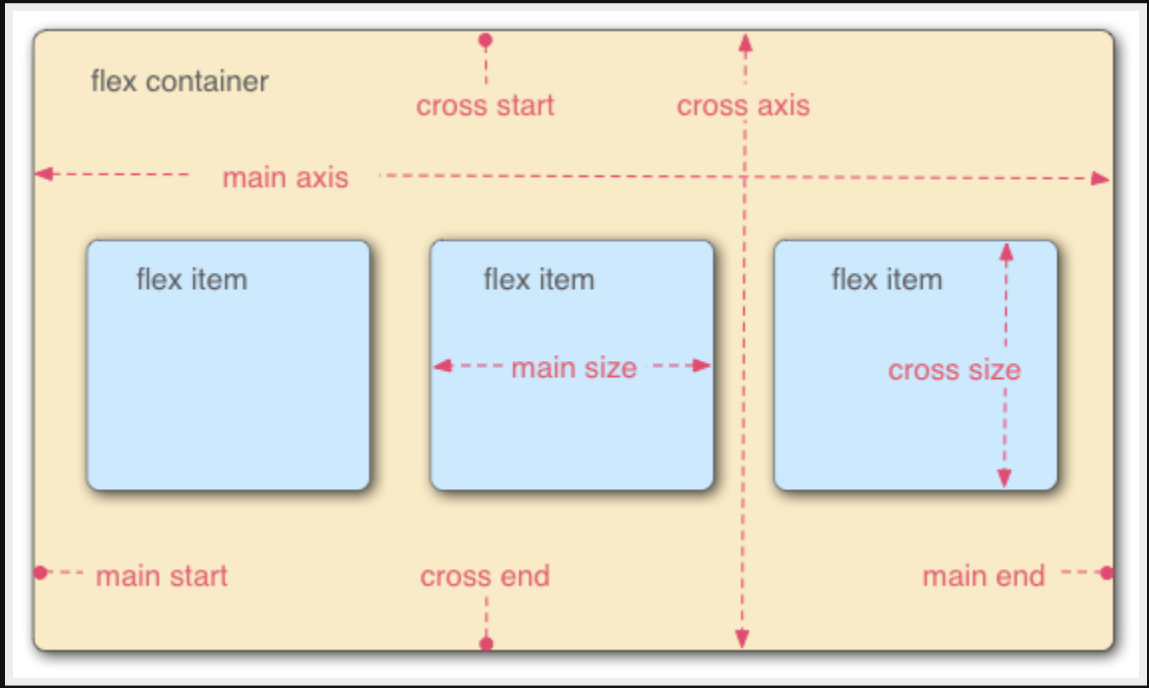
Konsep Dasar Flexbox

- Konsep dasar terbesarnya adalah kemampuannya untuk mengubah tinggi dan/atau lebar semua elemen-elemennya
- Sebuah kontainer flex akan mengubah ukurannya untuk mengisi ruang kosong atau memperkecilnya untuk mencegah overflow

Konsep Dasar Flexbox

- Algoritma flexbox tidak memperdulikan arah, berbeda dengan box layout yang bisa jadi horizontal (inline) atau vertikal (box)
- Box kurang cocok untuk perangkat dengan ukuran berbeda, orientasi berubah, dan seterusnya
- Flexbox cocok untuk diaplikasikan untuk komponen sebuah aplikasi dan layout dalam skala kecil

Istilah dalam Flexbox



Istilah dalam Flexbox

- Flex container: elemen induk di mana flex item berada. Ia didefiniskan dengan `flex` atau `inline-flex` sebagai nilai dari properti `display`
- Flex item: setiap anak dari sebuah flex container akan menjadi flex item.
- Axes: setiap layout flexbox mengikuti 2 sumbu, **main axis** adalah sumbu searah dengan arah para item disusun, **cross axis** adalah sumbu yang tegak lurus dengan **main axis**
  - `flex-direction` menentukan **main axis**
  - `align-items` menentukan bagaimana flex items disusun sepanjang cross axis

Istilah dalam Flexbox

- **main start/main end** dan **cross start/cross end** menetapkan titik awal dan titik akhir dari alur posisi flex item. Ini ditetapkan dalam `writing-mode`
- Ukuran dari sebuah flexbox ditentukan oleh **main size** dan **cross size**

Mengaktifkan Flexbox

- Untuk mengaktifkan flexbox, cukup gunakan properti CSS ini di div yang diinginkan sebagai kontainer sebuah flexbox

```
.flex-container {
    display: flex;
}
```

Contoh Kasus Flexbox: Menyusun Item Secara Horizontal

```
<style>
.flex-container {
    display: -webkit-flex;
    display: flex;
    -webkit-flex-direction: row;
    flex-direction: row;
}

.flex-item {
    margin: 10px;
}
</style>

<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
```

Contoh Kasus Flexbox: Menyusun Item Secara Vertikal

```
<style>
.flex-container {
    display: -webkit-flex;
    display: flex;
    -webkit-flex-direction: column;
    flex-direction: column;
}

.flex-item {
    margin: 10px;
}
</style>

<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
```

Contoh Kasus Flexbox: Menempatkan Elemen di Tengah

```
.flex-container{
    display: -webkit-flex;
   display: flex;
   -webkit-flex-direction: column; /* bisa row atau column */
   flex-direction: column; /* bisa row atau column */
   -webkit-align-items: center;
   align-items: center;
   -webkit-justify-content: center;
   justify-content: center;
    background-color: grey;
    height: 500px;
}
```

Contoh Kasus Flexbox: Mengatur Ukuran Flex Item

```
.bigitem {
   /* elemen ini ukurannya menjadi 2 kali elemen yang lebih kecil */
   -webkit-flex: 2 0 0;
   flex: 2 0 0;
}
.smallitem {
   -webkit-flex: 1 0 0;
   flex: 1 0 0;
}
```

Praktek

- Aplikasikan HTML dan CSS untuk halaman About dan Contact
- Silakan bekerja dalam tim, masing-masing tim terdiri dari 2 orang
- Font yang digunakan dalam referensi desain bisa diganti sesuai kebutuhan

Referensi Desain Halaman About

Cek di direktori

`project-resources/Mockup Website - About.png`

Referensi Desain Halaman Contact

Cek di direktori

`project-resources/Mockup Website - Contact.png`