

Kursus Coding Froyo Framework

Week 2: HTML dan CSS

HTML

Pengenalan HTML

- HTML, singkatan dari HyperText Markup Language
- *Markup language* standar untuk membuat halaman web
- Web browser bisa membaca file HTML dan me-render-nya menjadi halaman web
- HTML mendeskripsikan struktur sebuah website secara semantik, karena itu disebut sebagai *markup language* bukan *programming language*

Pengenalan HTML

- Elemen-elemen HTML adalah penyusun dari sebuah halaman HTML
- HTML memungkinkan gambar dan objek lainnya untuk dimasukkan ke dalam halaman web dan bisa digunakan untuk membuat form yang interaktif
- HTML juga mampu menciptakan dokumen yang terstruktur dengan memberikan elemen struktural seperti *heading*, paragraf, list, link, quote, serta yang lain
- Elemen HTML ditandai oleh tag <>
- Tag tidak ditampilkan browser, tapi digunakan untuk menginterpretasikan isi dari sebuah halaman web

Pengenalan HTML

- Sebuah script JS dan CSS bisa di-embed di dalam file HTML
- Ini bisa dilakukan untuk mendefinisikan tampilan dan layout teks dan elemen lainnya
- Praktek seperti ini lebih dianjurkan dibandingkan melakukan *styling* lewat tag HTML

HTML dan Browser

- Ketika me-*render* sebuah halaman HTML, browser akan membaca elemen-elemen serta *tag* HTML yang didefinisikan dalam file tersebut
- Dari situ, browser akan menampilkan halaman web, sesuai dengan struktur yang didefinisikan oleh file HTML
- Dengan demikian, untuk bisa menampilkan halaman web dengan benar, maka struktur halamannya harus benar dulu
- Kita bisa mulai dengan mendaftar dan mendefinisikan elemen-elemen apa saja yang harus ada di file HTML

Elemen HTML

- Elemen HTML adalah sebuah objek yang didefinisikan menggunakan *tag* HTML
- Elemen-elemen ini bisa berupa:
 - text: heading, paragraf, list
 - media: gambar, video, audio
 - layout: div, section, dsb.
 - form: button, form teks, menu dropdown, dsb.
- Secara otomatis, browser akan memberikan tempat bagi elemen-elemen ini di halaman web jika tag yang berkaitan ditemukan
- *Default*-nya, elemen yang akan di-*render* secara berurutan oleh browser, dan ditampilkan dari atas ke bawah

HTML5

- Versi terakhir dari HTML ini membawa beberapa *tag* yang bisa digunakan untuk membuat web yang lebih semantik, artinya elemen-elemen yang ada, jelas kedudukannya dalam layout
- Contoh:
 - footer
 - figcaption
 - section
 - dsb
- Selain itu, ia juga menyediakan beberapa elemen yang memungkinkan penggunaan media yang lebih beragam seperti canvas untuk animasi, webcam, audio, dsb

Penulisan HTML

- Secara umum, sebuah elemen HTML ditulis sebagai berikut:

```
<tag atribut="isi atribut">isiElemen</tag>
```

- Sebuah tag ditulis diawali dengan dan diakhiri dengan
- Sebuah tag bisa berisi atribut tertentu
- Isi sebuah elemen ditulis di antara dan

Contoh:

```
<a href="www.google.com">Link ke Google</a>
```

Struktur File HTML

- Sebuah File HTML dinyatakan dengan tag , karena ada *markup language* lain seperti XML
- Biasanya, file dibagi menjadi 2 bagian, dan
- `<head>` berisi metadata dari file HTML, bisa berisi:
 - judul halaman `<title>`
 - style CSS `<style>`
 - script JS `<script>`
 - dll.
- `<body>` mendefinisikan isi sebenarnya dari dokumen, di sinilah semua elemen dalam halaman didefinisikan

Struktur File HTML

```
<html>
<head>
  <title>Judul Halaman</title>
</head>

<body>
  Isi halaman
</body>

</html>
```

Heading dan Paragraf

- Menyediakan tingkat heading dari 1-6
- Sebuah paragraf bisa ditulis sebanyak-banyaknya baru kemudian ditutup

```
<h1>Heading 1</h1>  
<h2>Heading 1</h2>  
<h3>Heading 1</h3>  
<h4>Heading 1</h4>  
<h5>Heading 1</h5>  
<h6>Heading 1</h6>  
<p>Ini adalah kalimat awal sebuah paragraf. </p>
```

Link

- tagnya adalah
- atributnya berupa link yang dituju

```
<a href = "www.goal.com">Goal.com</a>
```

Gambar

- tagnya adalah ``
- atributnya adalah alamat gambar tersebut, caption dan deskripsi gambar (opsional)

```

```

List

- Bisa berupa list berurut atau tak berurut
 - Setiap elemen dalam list menggunakan tag
 -

```
<ol>
  <li>elemen no 1</li>
  <li>elemen no 2</li>
  <li>elemen no 3</li>
</ol>
<ul>
  <li>elemen tak berurut</li>
  <li>elemen tak berurut</li>
  <li>elemen tak berurut</li>
</ul>
```

Form

- Elemen yang digunakan untuk mengambil input dari user, baik berupa text, pilihan, berikut tombol submit

```
<form>
  Nama Lengkap: <input type="text" name="fullName">
  Jenis Kelamin: <select name="gender">
    <option value="pria">Pria</option>
    <option value="wanita">Wanita</option>
  </select>
  <input type="submit" value="Submit">
</form>
```


Tabel

- Sesuai namanya, bisa digunakan untuk membuat tabel
- Bisa juga digunakan untuk membuat layout, meski sekarang tidak dianjurkan
- Tagnya adalah , setiap baris dibuat dengan dan kolom dengan

```
<table>
  <tr>
    <td>Nama Depan</td>
    <td>Nama Belakang</td>
    <td>Gol</td>
  </tr>
  <tr>
    <td>Lionel</td>
    <td>Messi</td>
    <td>500</td>
  </tr>
</table>
```

Membagi Halaman ke dalam Bagian

- Sebuah halaman bisa dibagi ke dalam beberapa bagian, untuk kepentingan layout dan pengelompokan konten
- Kita bisa menggunakan tag
- Selain itu, ada juga tag lain yang memiliki arti khusus sesuai penggunaannya:
 - untuk elemen navigasi
 - untuk keseluruhan artikel
 - untuk gambar dalam artikel
- Biasanya, digunakan untuk menyatakan sebuah bagian halaman yang memiliki *style* CSS tersendiri
- Tag lain bisa digunakan untuk membagi halaman secara lebih semantik

- untuk sebuah bagian dalam artikel

Praktek HTML

Membuat Sitemap Website Project

Meninjau Referensi Desain Website Project

Membuat Struktur Halaman Home

Membuat Struktur HTML Dasar

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <header>
    </header>
    <section>
    </section>
    <footer>
    </footer>
  </body>
</html>
```

Membuat Judul Dokumen

```
<head>  
  <title>Galeri Instagram Kita</title>  
</head>
```


Menampilkan Judul Halaman

```
<header>
  <div>
    <h1>Galeri Kita</h1>
  </div>
```

Membuat Menu

```
<nav>
  <ul>
    <li><a href="">Home</a></li>
    <li><a href="">About</a></li>
    <li><a href="">Contact</a></li>
  </ul>
</nav>
```

Header Lengkap

```
<header>
  <div>
    <h1>Galeri Kita</h1>
  </div>
  <nav>
    <ul>
      <li><a href="">Home</a></li>
      <li><a href="">About</a></li>
      <li><a href="">Contact</a></li>
    </ul>
  </nav>
</header>
```

Menampilkan Gambar dan Caption Gambar

```
<section>
<div>
  <img src="">
  <p></p>
</div>
```

Membuat Footer

```
<footer>  
  <div>  
  </div>  
  <div>  
  </div>  
</footer>
```

Membuat Copyright di Footer

A Site by Loren Ipsum

Copyright©2016

Membuat Link Social Media di Footer

```
<div class="social-media-container">
  <ul>
    <li><a href = "http://www.facebook.com/blabla"></a></li>
    <li><a href = "http://www.instagram.com/blabla"></a></li>
    <li><a href = "http://www.twitter.com/blabla"></a></li>
  </ul>
</div>
```

CSS

Pengenalan CSS

- Cascading Style Sheet (CSS) digunakan untuk mengatur tampilan visual dari sebuah halaman web
- Termasuk di dalamnya:
 - layout halaman
 - posisi elemen
 - ukuran elemen
 - warna
 - font
 - dsb.
- Sesuai namanya, CSS pun tidak bisa dikategorikan sebagai sebuah bahasa pemrograman

CSS dan HTML

- CSS hanya bisa mengatur tampilan dari elemen HTML yang sudah ditentukan, di sini mereka berkaitan erat
- Pengaturan elemen yang satu akan mempengaruhi juga elemen yang di bawahnya
- Pastikan struktur HTML sudah benar, baru kita aplikasikan CSS

Penggunaan CSS

- Menggunakan CSS dilakukan dalam tahapan berikut:
 1. Memilih elemen HTML yang diinginkan -> CSS selector
 2. Mengisi parameter CSS yang sesuai -> CSS property
 3. Menutup deklarasi properti CSS untuk elemen tersebut

```
body {  
  width: 800px;  
  margin: 20px;  
}
```

Penggunaan CSS

- Setiap aturan CSS bagi sebuah elemen akan berlaku untuk semua elemen yang sama
- Contoh: sebuah header h1 yang sudah diatur ukurannya sekali, akan memiliki ukuran yang sama di seluruh dokumen HTML

```
/*CSS*/  
h1 {  
    font-size: 20px;  
}  
  
<!--HTML-->  
<!--kedua header memiliki ukuran sama-->  
<h1>Judul Halaman</h1>  
<h1>Nama Pengarang</h1>
```

Penggunaan CSS: id dan class

- Apabila kita ingin membedakan aturan CSS untuk 2 atau lebih elemen yang sama, maka kita bisa menggunakan id atau class
- id adalah sebuah identifier unik di dalam dokumen.
 - Penggunaannya berkaitan dengan isi dari sebuah elemen
- class adalah sebuah identifier yang bisa digunakan untuk beberapa elemen di dalam dokumen
 - Penggunaannya berkaitan dengan styling dari sebuah elemen
- Keduanya didefinisikan di dalam tag HTML

```
/*CSS untuk class*/
h1 {
    font-size: 20px;
}
h1 .small{
    font-size: 15px;
}

<!--HTML-->
<!--kedua header memiliki ukuran sama-->
<h1>Judul Halaman</h1>
<h1 class="small">Nama Pengarang</h1>
```



```
/*CSS untuk id*/
h1 {
    font-size: 20px;
}
h1 #nama{
    font-size: 15px;
}

<!--HTML-->
<!--kedua header memiliki ukuran sama-->
<h1>Judul Halaman</h1>
<h1 id="nama">Nama Pengarang</h1>
```

Penggunaan CSS: Urutan Penulisan

- Sebuah aturan CSS bisa di-*override* berkali-kali di dalam satu file CSS
- Aturan CSS yang berlaku adalah yang paling terakhir ditulis

```
h1 {  
  color: #dd0000;  
}  
h1 {  
  color: #00dd00;  
}
```

Penggunaan CSS: Ukuran

- Untuk mengatur ukuran dalam CSS, kita bisa menggunakan satuan-satuan berikut:
 - persen (%): ukuran adalah presentase dari lebar halaman
 - pixel (px): ukuran adalah sekian pixel
 - em: ukuran relatif terhadap ukuran font dari elemen tersebut
 - rem: ukuran relatif terhadap ukuran font dari elemen root
- Ada banyak satuan lain, tapi keempat itu adalah yang paling umum

Penggunaan CSS: Warna

- Warna dalam CSS diatur dengan beberapa notasi
- Aturannya adalah kita mengatur warna berdasarkan intensitas pada channel red, green dan blue (RGB)
- Masing-masing warna memiliki nilai 0-255
- Ketika ditulis dalam notasi HEX (hexadecimal), nilainya menjadi 00-FF
- Untuk menyederhanakan, kita bisa menggunakan salah satu dari notasi-notasi berikut:
 - HEX: contoh `#ff0000` -> nilai red maksimum, green dan blue 0.
 - RGB: contoh `rgb(255, 0, 0)`
 - RGBA: contoh `rgba(255, 0, 0, 20)`

Penggunaan CSS: Layout

- Perlu dipahami bahwa secara default, semua elemen HTML akan memkai 100% lebar sebuah halaman
- Elemen ini kemudian akan di-*render* secara berurutan oleh browser, hingga nampak berurutan dari atas ke bawah
- Untuk mengatur posisi dari sebuah elemen, maka kita bisa mengatur ukuran lebarnya, *alignment-nya*, serta margin dan paddingnya

- Kita punya struktur HTML berikut, dan kita ingin agar keduanya bersebelahan

```
<div id="nav-menu">
  <ul>
    <li>Home</li>
    <li>About</li>
    <li>Contact</li>
  </ul>
</div>

<div id="main-content">
  <p>Ini adalah isi utama artikel. Menu bisa dilihat di samping kiri saya</p>
</div>
```

Penggunaan CSS: Strategi Layout

- Untuk memenuhi tujuan kita, maka kita bisa melakukan langkah-langkah berikut:
 - Mengatur ukuran elemen `nav-menu` sehingga tidak memakan 100% lebar halaman. Dicapai dengan mengatur `width`-nya
 - Mengatur agar `main-content` berdiri di kiri `nav-menu`. Dicapai dengan mengatur `float`-nya

```
div #nav-menu {  
  width: 300px;  
}  
  
div #main-content {  
  float: left;  
  width: 500px;  
}
```


Penggunaan CSS: Margin dan Padding

- Margin dan Padding adalah 2 aturan CSS yang bisa kita gunakan untuk mengatur posisi sebuah elemen
- Sejatinya, keduanya bekerja dengan mengatur jarak antara sebuah elemen HTML, dengan elemen di dekatnya
- Jaraknya bisa diatur untuk posisi atas, bawah, kiri atau kanan
- `margin` digunakan untuk mengatur jarak antara 1 elemen dengan elemen sesudah dan sebelumnya
- `padding` mengacu pada jarak antara elemen induk dengan elemen anaknya

```
<!--HTML-->
<div id="pertama">
    <p>Ini adalah elemen pertama</p>
</div>
<div id="kedua">
    <p>Ini adalah elemen kedua</p>
</div>
```

```
/*CSS*/
/*Jarak antara pertama dan kedua adalah 10px*/
/*Jarak antara paragraf dalam kedua adalah 20 px*/
div #pertama {
    margin: 10px;
}
div #kedua {
    padding: 20px;
}
```

Praktek

Framework CSS

- Website kita banyak menggunakan Grid untuk membuat layout image di dalam gallery
- Menggunakan CSS manual bisa, namun akan menghabiskan cukup banyak waktu
- Kita bisa menggunakan CSS framework, yang berisi library yang akan mempermudah membuat layout yang berdasarkan grid
- Kita akan menggunakan PureCSS, karena ukurannya kecil, namun fiturnya cukup lengkap

Instalasi PureCSS

```
<head>
```

```
<link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/pure-min.css">
```

Mengenal Grid

- Grid adalah sebuah sistem visual, di mana sebuah halaman dibagi ke dalam beberapa kolom
- Kolom ini lebarnya bisa sama ataupun tidak
- Gambar dalam website kita disusun menggunakan grid, di mana semuanya memiliki lebar dan jarak yang sama di antaranya
- Kita perlu sedikit menghitung bagaimana mengaplikasikan grid untuk project kita

Grid di Project

- Semua konten gambar disimpan dalam sebuah container berukuran tetap
- Grid kemudian diaplikasikan di sini, di mana kontainer akan dibagi 3 sama besar
- Semua gambar beserta captionnya akan disimpan dalam kontainer yang telah dibagi tadi

Grid dengan PureCSS

- Menggunakan PureCSS, kita cukup memanggil kelas untuk grid yang sesuai
- PureCSS memiliki grid dalam 2 satuan: $1/24$ dan $1/5$, berlaku kelipatannya, seperti: $5/12$, $1/2$, $4/5$, dst.
- Setiap jenis grid ini sudah memiliki kelas sendiri yang tinggal kita gunakan


```
<div class="pure-u-8-12 image-container" >
  <div class="pure-u-1-3">
    
    <p class="caption">Dummy text</p>
  </div>
  <div class="pure-u-1-3">
    
    <p class="caption">Dummy text</p>
  </div>
  <div class="pure-u-1-3">
    
    <p class="caption">Dummy text</p>
  </div>
</div>
```

Grid dengan PureCSS

- Untuk menggunakan class grid, maka induk dari elemen yang akan diatur, dalam hal ini `<section>`, perlu diberi class "pure-g", menandakan bahwa kita akan memanggil grid PureCSS untuk semua elemen anaknya
- Ketiga gambar diletakkan dalam sebuah kontainer
- Kontainer tidak berukuran penuh, melainkan hanya 8/12 (=75%) dari lebar halaman total. Ini dicapai dengan menggunakan pure-u-8-12
- Kontainer ini kemudian dibagi 3 sama rata dengan menggunakan class pure-u-1-3

Grid dengan PureCSS

- Perhatikan bahwa kita menambahkan 1 lagi `<div>` dengan class `image-container` untuk membungkus 3 gambar dalam 1 baris di dalam gallery.
- Teknik ini berarti kita menggunakan `div` sebagai alat bantu layout
- Dengan menggunakan ini, maka kita bisa mengatur lebar kontainer dan kita bisa mengatur posisinya juga

Mengatur Posisi Kontainer Grid

- Untuk memastikan agar kontainer bisa diletakkan di tengah halaman, maka kita bisa mengatur nilai `margin`-nya serta menetapkan `width`-nya
 - Dilakukan dengan menggunakan `margin: auto`, otomatis ia akan ditempatkan di tengah terhadap elemen induknya (dalam hal ini, `<section>`)
- Pengaturan ini dilakukan dengan memilih `class` dari kontainer tersebut, lalu membuat aturan CSS-nya

Mengatur Posisi Kontainer Grid

```
.image-container {  
  margin: auto;  
  margin-top: 60px;  
  margin-bottom: 50px;  
  width: 75%;  
}
```

Mengatur Gambar dalam Grid

- Kita perlu melakukan 2 hal:
 - Mengatur ukuran gambar
 - Membuatnya berada di tengah grid miliknya

Mengatur Gambar dalam Grid

```
img {  
  max-width: 300px;  
  display: block;  
  margin-right: auto;  
  margin-left: auto;  
}
```

- `display: block` dilakukan agar gambar bisa menggunakan keseluruhan lebar grid miliknya sehingga posisinya lebih teratur

Mengatur Caption Gambar

- Karena caption sudah kita berikan `class` sendiri, maka ia bisa kita atur sehingga posisinya bisa sesuai di dalam grid
- Elemen `<p>` memiliki kecenderungan untuk memaksa menambahkan lebar tempat ia berada, sehingga kita perlu mengatur lebar maksimalnya menggunakan `max-width`
- Dalam hal ini, lebar maksimalnya kita samakan dengan ukuran gambar

Mengatur Caption Gambar

```
p.caption {  
  max-width: 300px;  
  margin: auto;  
  text-align: left;  
  margin-top: 10px;  
}
```