HTML & CSS Basics - First Summary

Kami belajar banyak tentang HTML dan CSS - dua bahasa terpenting yang perlu Anda ketahui sebagai pengembang web.

Berikut adalah ringkasan singkat dari semua fitur inti yang kami pelajari sejauh ini. Fitur yang Anda **perlu tahu** karena kamu akan **gunakan setiap saat!**

HTML Summary

What & Why?

adalah jantung dari setiap halaman web. Ini mendefinisikan struktur halaman web dan membubuhi keterangan konten untuk memberi tahu browser apa yang harus ditampilkan dan memberikan makna tambahan. Tanpa HTML, teks akan selalu menjadi "hanya teks"
- tidak akan ada perbedaan semantik antara judul, teks normal, subtitle, dll. Anda juga tidak akan dapat menambahkan gambar atau tautan.

Meskipun browser secara default menambahkan gaya untuk elemen HTML tertentu (mis `<h1>` elemen tebal dan lebih besar secara default), Anda **jangan gunakan HTML untuk tujuan penataan!**

Sebaliknya, HTML menambahkan makna semantik ke konten Anda - HTML memberikan anotasi yang menjelaskan konten Anda. Dan dalam kasus elemen tertentu (mis `<img>`, `<a>`) juga memberitahu browser apa yang akan ditampilkan (misalnya menampilkan gambar) dan apa yang harus dilakukan (misalnya menavigasi ke halaman lain).

Situs web Anda tidak hanya dilihat oleh Anda tetapi juga diuraikan oleh perayap mesin telusur atau disajikan oleh teknologi bantu seperti pembaca layar. Itulah mengapa mendeskripsikan konten Anda dengan benar sangat penting!

HTML Element Anatomy

Elemen HTML biasanya terdiri dari tag pembuka(mis `<h1>`), beberapa **content** (mis `Hi there!`) dan **tag penutup** (mis `</h1>`).
Elemen HTML juga dapat menerima **atribut** yang memungkinkan Anda untuk menambahkan perilaku atau konfigurasi ekstra ke elemen (tergantung pada atributnya).

Misalnya, tautan (`<a>`) membutuhkan `href` untuk memberi tahu browser ke mana ia harus mengarah:

```
<p>This leads <a href="https://www.google.com">to Google</a>.</p>
```

Ada juga beberapa **void element** - misalnya `<img>`. Elemen-elemen tersebut tidak memerlukan tag penutup karena tidak memiliki konten apa pun.

```
<img src="some-image.jpg" alt="A beautiful image of something">
<!-- or -->
<img src="some-image.jpg" alt="A beautiful image of something" />
```

Nesting HTML Elements / Elemen HTML Bersarang

Salah satu karakteristik utama HTML adalah Anda dapat menyatukan elemen satu sama lain - seperti yang dapat Anda lihat dalam contoh ini:

```
<ul>
  <li>This leads to <a href="https://www.google.com">Google</a></li>
  <li>Another item</li>
</ul>
```

Di sini, `<li>` elemen yang disebut **child element** dari `<ul>` (yaitu **parent**). Dan bahkan memiliki elemen anak itu sendiri: Tautan (`<a>`) elemen (yaitu **turunan dari** elemen `<ul>` di sisi lain, `<ul>` adalah **ancestor/leluhur** dari `<a>`).

Kedua `<li>` ("Another item") akan menjadi **sibling** ke yang pertama `<li>` – Bersama-sama, mereka membentuk **children** dari `<ul>`.

Saat menulis kode HTML, Anda biasanya membangun struktur yang sangat bersarang karena sebagian besar konten hanya dapat dijelaskan secara akurat jika Anda menggabungkan elemen HTML seperti ini. Bagaimana lagi Anda akan membuat daftar tautan yang benar secara semantik?

HTML Structure & Skeleton

Ada dua kelompok besar elemen HTML:

1. Elemen untuk menampilkan dan mendeskripsikan konten halaman Anda (mis `h1`, `p`, `ul`, `a`, `img` etc.)
2. Elemen untuk menjelaskan keseluruhan halaman Anda dan untuk menautkan ke sumber daya lain yang diperlukan (mis `title`, `meta`, `link`, `style`)

Itu sebabnya dokumen HTML yang diformat dengan benar harus memiliki "skeleton/kerangka" yang mendefinisikan dua area utama:

1. `<body>` untuk konten halaman
2. `<head>` untuk metadata

Oleh karena itu, kerangka dokumen HTML yang benar (yang harus Anda gunakan di setiap `.html` file yang Anda buat) akan terlihat seperti ini:

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Page Title</title>
    <!-- Other metadata -->
  </head>
  <body>
    <h1>Hi there!</h1>
    <!-- Other page content -->
  </body>
</html>
```

`<!-- .... -->` menandai komentar ini tidak akan muncul di halaman saat diberikan oleh browser.

Inline vs Block Elements

You also learned that are two main kinds of content (`<body>`) elements in HTML:

1. Block elements (mis `h1`, `p`, `ul`, ...)
2. Inline elements (mis `a`, `span`, `img`)

Elemen blok selalu mencadangkan seluruh lebar layar untuk dirinya sendiri (meskipun itu dapat diubah oleh CSS).

Elemen inline di sisi lain - seperti namanya - ada untuk menyesuaikan "ke dalam garis dengan elemen lain". Misalnya, akan aneh jika tautan (`<a>`) akan memaksa jeda baris karena menyimpan seluruh lebar layar untuk dirinya sendiri.

Sebaliknya, paragraf baru (`<p>`) atau judul (`<h1>`, `<h2>` etc.) mungkin tidak boleh dimasukkan ke dalam baris yang sama dengan beberapa teks lainnya.

Anda dapat mengubah perilaku itu melalui CSS, jika perlu.

HTML Element yang Kita Ketahui Sejauh ini

Hingga saat ini, kami telah mempelajari banyak elemen penting - berikut daftar ringkasannya:

- `<head>` Elements
    - `<meta>`: Dapat digunakan untuk menambahkan metadata tambahan ke halaman Anda - misal deskripsi yang dapat diambil oleh perayap mesin telusur
    - `<title>`: Memungkinkan Anda untuk menentukan judul halaman (akan digunakan oleh mesin pencari tetapi juga muncul di tab browser)
    - `<style>`: Dapat digunakan untuk menentukan gaya CSS global untuk halaman
    - `<link>`: Memungkinkan Anda untuk menautkan ("connect") file HTML Anda ke beberapa sumber lain - biasanya ke `.css` file yang gayanya kemudian akan diterapkan sebagai gaya global ke dokumen HTML itu

- `<body>`Elements:
    - `<h1>`, `<h2>`, `<h3>` etc: Headings (titles) yang dapat Anda tambahkan ke halaman Anda. Harus digunakan secara berurutan dan hanya satu `<h1>` elemen harus digunakan per dokumen HTML. `<h2>` elemen bertindak sebagai subjudul, `<h3>` elemen kemudian subjudul lain pada tingkat yang lebih rendah.
    - `<p>`: Dapat digunakan untuk mendefinisikan paragraf reguler dari konten sebaris (biasanya teks)
    - `<em>`: Dapat digunakan untuk menambahkan penekanan pada kata atau frasa

- `<strong>`: Dapat digunakan untuk menambahkan penekanan yang kuat pada kata atau frasa (seolah-olah Anda akan mengatakannya dengan keras, sangat menekankannya secara verbal)
- `<img>`: Dapat digunakan untuk menampilkan gambar. Anda perlu mengatur jalur sumber gambar melalui `src` atribut. Dan Anda harus mengatur beberapa teks alternatif melalui alternatif atribut - teks ini ditampilkan jika gambar tidak dapat dimuat atau jika pengguna yang menggunakan teknologi bantu mengunjungi halaman Anda

- `<ul>`/`<ol>`: Dapat digunakan untuk membuat daftar data yang berurutan atau tidak berurutan. Gunakan elemen ini jika Anda mendapatkan data daftar.
- `<li>`: Digunakan di dalam `<ul>` atau `<ol>` untuk menentukan item daftar individual dari daftar itu.
- `<header>`: Elemen blok yang mendefinisikan header - untuk seluruh halaman atau subbagian halaman (lebih lanjut tentang itu nanti)
- `<footer>`: Elemen blok yang mendefinisikan footer - untuk seluruh halaman atau subbagian halaman (lebih lanjut tentang itu nanti)
- `<main>`: Mendefinisikan konten utama halaman - Anda tidak boleh memiliki lebih dari satu `<main>` elemen per dokumen HTML.
- `<section>`: Mendefinisikan bagian baru dalam dokumen Anda - biasanya berisi heading (mis `<h2>`), meskipun itu bukan "harus dimiliki"
- `<span>`: Elemen sebaris yang tidak berarti yang dapat digunakan untuk membungkus konten yang harus ditargetkan dengan CSS style (misalnya dengan bantuan class atau id CSS)
- `<div>`: Elemen blok tidak berarti yang dapat digunakan untuk membungkus konten yang harus ditargetkan dengan CSS style (misalnya dengan bantuan class CSS atau id)

Kami tentu saja akan melihat lebih banyak elemen sepanjang kursus ini - Anda juga dapat menjelajahi semua elemen yang tersedia di MDN HTML Elements Reference.

CSS Summary

What & Why?

CSS (Cascading Style Sheets) adalah bahasa yang memungkinkan Anda untuk menentukan definisi gaya untuk dokumen HTML Anda. Definisi ini akan diambil oleh browser dan mengontrol bagaimana konten akan ditampilkan (mis beberapa teks berwarna merah, ukuran font yang seharusnya, dll.).

Di mana HTML mendefinisikan struktur dan makna konten Anda (dan tentu saja juga membantu menampilkan konten di tempat pertama - misalnya dengan elemen `<img>`), CSS digunakan untuk kemudian menyajikan konten persis seperti yang

Anda inginkan untuk menyajikannya. Itu menambahkan **tidak ada artinya** atau anotasi - ini benar-benar hanya tentang bagaimana segala sesuatunya terlihat.

Jika anda ingin mengubah warna teks, warna latar belakang, ukuran, spasi, jarak, batas, bayangan, posisi, dan semua hal ini, Anda memerlukan CSS.

CSS Syntax

Ada berbagai macam CSS yang tersedia **properti** yang dapat Anda definisikan. Semua properti itu kemudian mengambil yang berbeda **value** - nilai konkret yang dapat Anda tetapkan bergantung pada properti yang Anda tetapkan.

```
color: #ccc; /* hexadecimal represention of rgb color */
color: rgb(204, 204, 204); /* same colors as above */
font-size: 18px; /* setting a size in device-independent pixels */
```

CSS Style dapat didefinisikan dengan cara yang berbeda, misalnya "sebaris" melalui style atribut:

```
<h1 style="font-size: 20px">Hi there!</h1>
```

Tetapi menggunakan "gaya sebaris" seperti itu biasanya tidak disarankan karena memiliki berbagai kelemahan. Yang terpenting, ini mengacaukan kode HTML Anda (sulit dirawat!), Membutuhkan banyak duplikasi dan salin dan tempel dan juga sulit untuk diubah: Anda sering harus mengunjungi dan menyesuaikan beberapa elemen jika Anda ingin menyesuaikan satu single style.

Itu sebabnya anda biasanya mendefinisikan **global CSS styles** melalui `<style>` elemen atau di eksternal `.css` file yang kemudian diimport melalui `<link>` elemen.

Saat menggunakan gaya global seperti itu, sintaks CSS tetap sama tetapi Anda mengelompokkan penetapan nilai properti Anda ke dalam apa yang disebut **CSS rule** yang juga **CSS selector** yang menentukan untuk elemen HTML mana gaya Anda harus diterapkan:

```
h1 { /* is applied to ALL <h1> elements */
  font-size: 20px;
}

#some-id { /* is applied to a single HTML element with id="some-id" */
  color: #ccc;
}

.my-link { /* is applied to ALL HTML elments with class="my-link" */
  text-decoration: none;
}
```

Anda juga dapat menggabungkan CSS selectors:

```
h1, p { /* applies the rule to all <h1> and <p> elements */
  color: rgb(204, 204, 204);
}

p a { /* targets all <a> elements that are descendants of a <p> */
  color: red;
}
```

Anda juga dapat memberikan gaya khusus untuk status HTML element tertentu - dengan pseudo-selectors:

```
a:hover { /* targets <a> where the user's mouse is hovering over */
  font-weight: bold;
}
```

What Can You Style?

Anda benar-benar dapat menata (hampir) apa saja dengan CSS. Itu sebabnya Anda harus menggunakan CSS untuk penataan dan tidak menggunakan elemen HTML tertentu karena Anda menginginkan tampilan tertentu.

HTML ada untuk mendefinisikan konten, memberikan struktur dan makna, CSS kemudian ada untuk menyajikan konten Anda namun Anda ingin menyajikannya.

CSS Values & Units

Karena ada lusinan properti CSS yang dapat Anda targetkan (sejauh ini kami hanya melihat sebagian kecil), ada juga ratusan kemungkinan nilai yang dapat Anda tetapkan.

Nilai konkret yang dapat Anda tetapkan selalu bergantung pada properti yang sedang Anda definisikan: Color properties (mis `color`, `background-color`) menginginkan nilai warna (mis `#fa923f`, `#ccc`, `rgb(204, 204, 204)`, `hsl(180, 20, 75)`, `red`). Dimension properties (mis `font-size`, `margin`) menginginkan nilai dimension (mis `18px`). `font- family` property daftar font family yang harus digunakan (mis `font-family: 'Oswald', sans-serif`).

Meskipun jumlah properti dan nilai serta unit yang tersedia bisa jadi menakutkan saat ini, Anda akan mengembangkan perasaan yang baik untuk properti dan nilai yang berbeda dari waktu ke waktu. Pada akhirnya, Anda akan selalu memiliki beberapa properti dan nilai yang sangat sering Anda gunakan (dan karena itu hafal).

Tentu saja Anda juga selalu dapat menggunakan dukungan dan sumber daya pelengkapan otomatis IDE seperti MDN CSS Properties List untuk mempelajari semua sifat dan nilai yang mungkin.

The Box Model

Saat bekerja dengan CSS, penting untuk dipahami bahwa elemen HTML memiliki apa yang disebut "box model".

Ini berarti bahwa semua elemen HTML memiliki berbagai "lapisan" yang dapat ditata:

- konten
- `padding` disekirat konten (tetapi di dalam batas)
- `border`
- `margin` di sekitar seluruh elemen

Anda tidak perlu mengatur apapun `padding`, `border` atau `margin` - Anda hanya dapat mengatur apa yang Anda butuhkan. Tetapi dengan "layers" yang berbeda ini, Anda dapat menambahkan jarak antara batas yang terlihat dan konten dan / atau di sekitar elemen.

Perlu dicatat bahwa elemen blok dan inline berperilaku berbeda:

- **Block elements**: berperilaku sepertia yang dijelaskan di atas
- **Inline elements**: Vertikal `margin` diabaikan, vertical `padding` ditambahkan tetapi tidak mendorong elemen atau konten lain menjauh.

Anda juga dapat mengubah jika suatu elemen berperilaku seperti blok atau elemen sebaris melalui `display` property (mis `display: block` menyetel perilaku ke "block elemen" meskipun biasanya merupakan inline elemen). Disana juga `inline-block` mungkin `display` nilai yang pada dasarnya akan menggabungkan dua

perilaku dan "membuka" perilaku blok penuh sambil menjaga konten tetap inline dengan konten lain.

Why Is It Called "Cascading"?

Mengapa disebut "Cascading Style Sheets"?

Karena serangkaian definisi gaya dapat memengaruhi satu dan elemen yang sama - yaitu beberapa aturan CSS dapat memengaruhi elemen yang sama.

Berikut ini contohnya:

```
<style>
  a {
    text-decoration: none;
  }

  .default-link {
    color: red;
  }
</style>

<a class="default-link" href="...">Some linke</a>
```

Dalam contoh ini, kedua aturan CSS memengaruhi elemen `<a>`.

Mengapa Anda memiliki dua, bukan hanya satu aturan? Karena Anda mungkin memiliki tautan yang berbeda di halaman Anda - beberapa dengan `default-link` class, beberapa tanpa itu.

Karena beberapa aturan CSS dapat memengaruhi elemen HTML yang sama, Anda dapat membagi definisi CSS menjadi beberapa aturan. Ini menghindari copy & paste dan duplikasi kode yang tidak perlu!

Specificity

Tetapi apa yang terjadi jika aturan CSS yang berbeda menargetkan CSS

property yang sama? Pertimbangkan contoh ini:

```
<style>
  a {
    color: blue;
  }

  .default-link {
    color: red;
  }
</style>

<a class="default-link" href="...">Some linke</a>
```

Dalam hal ini, tautan `red` karena urutan CSS rules penting – aturan selanjutnya (dalam hal ini `.default-link` menang).

Tapi sebenarnya, ini sedikit lebih kompleks dari itu. Ini bukan hanya tentang

pesanan.

Sebaliknya, ada konsep yang disebut **specificity**.

Semakin tinggi spesifisitas suatu aturan, semakin penting aturan tersebut. Dan spesifisitas yang lebih tinggi mengalahkan spesifisitas yang lebih rendah.

Urutan kode Anda mempengaruhi specificity tetapi itu bukan satu-satunya faktor.

Jenis pemilih yang Anda gunakan juga memainkan peran penting - misalnya, id

selector (seperti `#some-id`) mengalahkan regular tag type selector (seperti `h1`).

Dan itu masuk akal: Lagi pula, jika Anda menargetkan id tertentu, Anda memiliki kriteria pemilihan yang jauh lebih spesifik daripada jika Anda hanya menargetkan jenis tag.

Dan sementara saya sekarang bisa menulis daftar panjang faktor kekhususan dan aturan mana yang menang atas aturan lain, hal yang menghibur adalah bahwa pada dasarnya Anda dapat mempercayai akal sehat Anda sendiri.

Sama seperti masuk akal bahwa id lebih spesifik daripada "hanya tag", combinasi selector (mis `p a { ... }`) akan memenangkan "standalone" selector (mis `a { ... }`).

Jadi, semakin spesifik pemilih CSS Anda, semakin tinggi spesifisitasnya. Dan spesifisitas yang lebih tinggi mengalahkan spesifisitas yang lebih rendah.

Inheritance

Terhubung dengan konsep "Cascading" dan konsep "Specificity" adalah konsep dari "Inheritance".

Elemen HTML tidak hanya dipengaruhi oleh gaya yang didefinisikan dalam aturan yang secara langsung menargetkan elemen tersebut, tetapi juga dapat ditata oleh aturan yang menargetkan elemen induk atau leluhur.

Contohnya :

```
<style>
  body {
    font-family: 'Open Sans', sans-serif;
    text-align: center;
  }

  p {
    margin: 20px;
  }
</style>

<body>
  <h1>I use 'Open Sans' as a font-family!</h1>
  <p>So do I! And we're both aligned to the center.</p>
</body>
```

Dalam contoh ini, keduanya `<h1>` dan `<p>` akan terpengaruh oleh `font-family` dan `text-align` definisi CSS di `body` rule.

Why?

karena "Inheritance/pewarisan".

Mereka mewarisi style yang ditentukan elemen parent.
Tidak semua properti CSS dapat diwariskan - tetapi sekali lagi, Anda dapat mempercayai akal sehat Anda.

Text color, Sebagian besar style font dll. Adalah

pewarisan. Margin, padding, border dll. tidak.

Browser Defaults

Browser juga menentukan beberapa gaya default untuk elemen tertentu - misalnya, mereka sering membuat elemen `<h1>` lebih besar dan memberikan `bold font-weight`.

Ini memiliki spesifisitas yang sangat rendah dan karenanya Anda dapat dengan mudah menimpa default styles ini dengan CSS selector apapun.

CSS Properties yang Kami Ketahui Sejauh ini

- Font-related:
  - `font-family:` Atur jenis font yang ingin Anda gunakan - bisa berupa font tunggal atau (biasanya) daftar font, di mana font pertama digunakan dan font lainnya bertindak sebagai font cadangan jika font pertama tidak dapat dimuat
  - `font-size`: Mengatur ukuran teks - dapat diatur dalam piksel "perangkat independen" (mis `18px`) atau nilai lain yang akan kita temukan nanti
  - `font-weight`: Memungkinkan Anda untuk weight bobot beberapa teks (default adalah `normal`, Anda dapat memilih numeric weight seperti `700` atau alternatif seperti `bold`). Penting: Font yang Anda muat harus mendukung weight itu!
- Text:
  - `text-align`: mengontrol perataan teks (mis `center`, `left`, `right`)
  - `text-decoration`: Digunakan untuk menambahkan hiasan tambahan (atau menghilangkannya) seperti underlining (mis `text-decoration: underline`)
- Colors:
  - `color`: Mengatur warna teks
  - `background-color`: Mengatur warna latar belakang elemen
  - warna dapat diatur dengan berbagai jenis unit:

        - Hexadecimal number identifiers (mis `#fa923f`, `#ccc`) di mana Anda mendapatkan tiga pasangan dua digit yang menentukan bagian warna r/g/b (merah, hijau, biru)
        - `rgb()` "function" yang memungkinkan Anda menyetel warna ar/g/b dengan angka desimal
        - `hsl()` fungsi yang dapat digunakan untuk mendefinisikan warna sebagai kombinasi rona/saturasi/ringan
        - Anda dapat menggunakan salah satu dari unit warna ini - tergantung pada preferensi pribadi Anda; setiap warna dapat diekspresikan dengan masing-masing dari tiga metode ini
        - Ada juga `rgba()` dan `hsla()` jika Anda juga ingin menambahkan "alpha channel" (transparansi - nilai antara 0 dan 1) ke warna Anda
- Box Model:
    - `margin`: Mengatur jarak ekstra **di sekitar elemen**-dapat diatur dengan pixels (mis `18px`)
        - `margin` adalah notasi singkatan yang mengatur jarak ke segala arah, bentuk panjangnya adalah `margin: <top> <right> <bottom> <left>` (mis `margin: 10px 5px 8px 3px`)
        - Anda juga punya `margin-left`, `margin-right`, `margin-top` `margin-bottom` untuk menentukan target tertentu
    - `padding`: menambah extra spasi **didalam element** – dapat diatur dengan pixel (mis `18px`)
        - Seperti `margin`, Anda dapat menggunakan shorthand notation `padding: <top> <right> <bottom> <left>` atau target spesifik tertentu seperti `padding-bottom`
    - `border`: Can be used to define a visible border around the element content + padding (e.g. `border: 1px solid black`)
    - `border-radius`: Dapat digunakan untuk memberikan sudut kotak yang dibulatkan (walaupun tidak ada batas yang ditentukan)
- Sizes:
    - `width`: Digunakan untuk menentukan lebar tetap untuk suatu elemen - alih-alih menggunakan lebar layar penuh untuk elemen blok atau lebar konten untuk inline elemen (dapat diatur dalam piksel seperti `18px`)
    - `height`: Digunakan untuk menentukan ketinggian tetap untuk suatu elemen - alih-alih menyimpulkannya secara otomatis berdasarkan tinggi konten dalam elemen
- Other:
    - `box-shadow`: Dapat digunakan untuk mendefinisikan bayangan untuk suatu elemen - diatur dengan mendefinisikan x-offset, y-offset, radius blur optional, radius penyebaran opsional, dan warna (mis `box-shadow: 0 1px 8px rgba(0, 0, 0, 0.2)`)