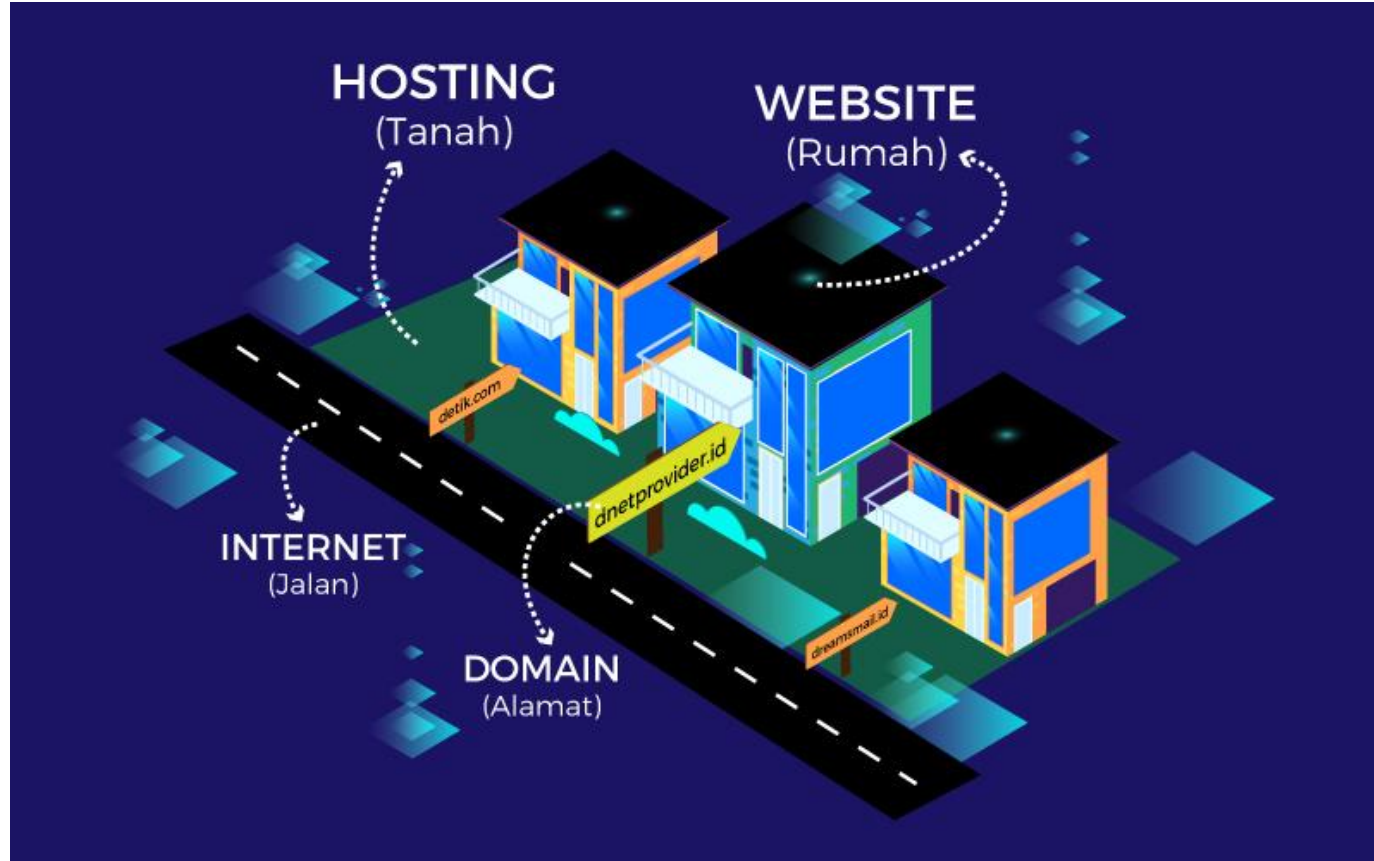
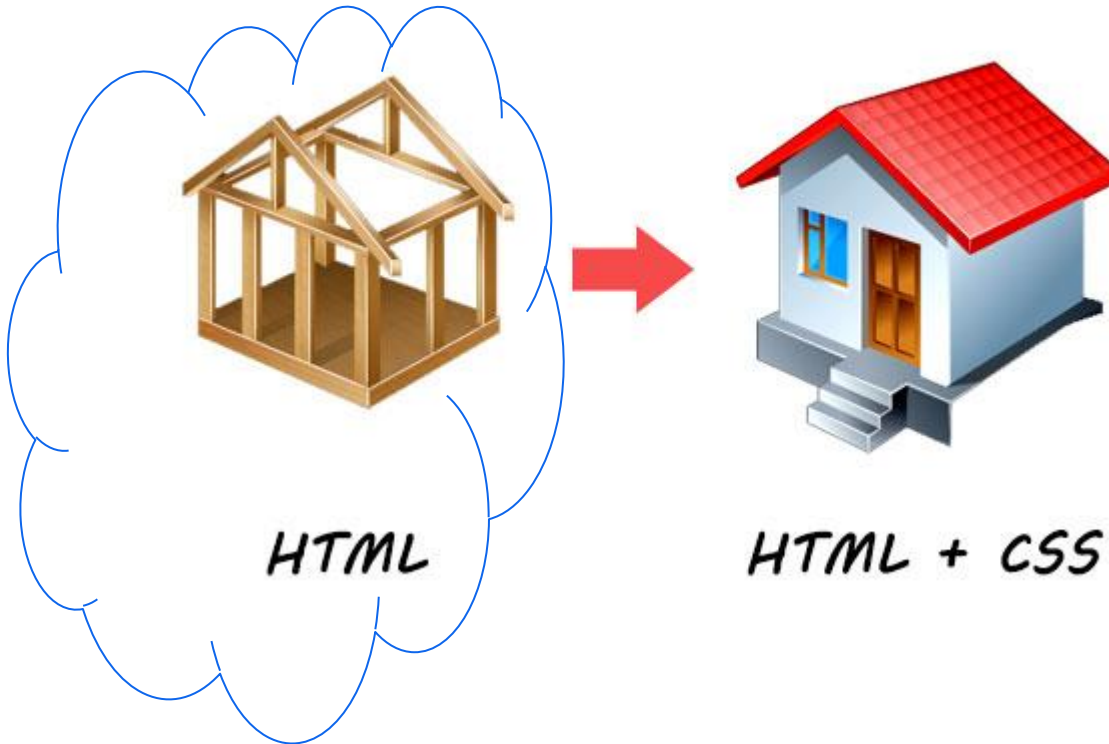


Review w1 : Analogi Web



Maka di w2 : Analogi Website Statis



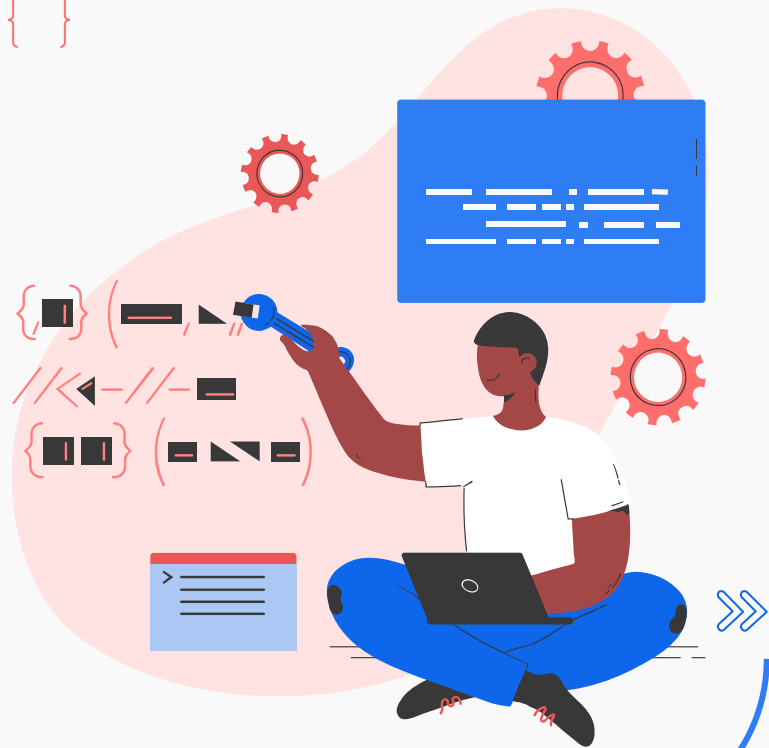
Html & validasi + git

Pemrograman Web #2



01

HTML



HTML : Definisi

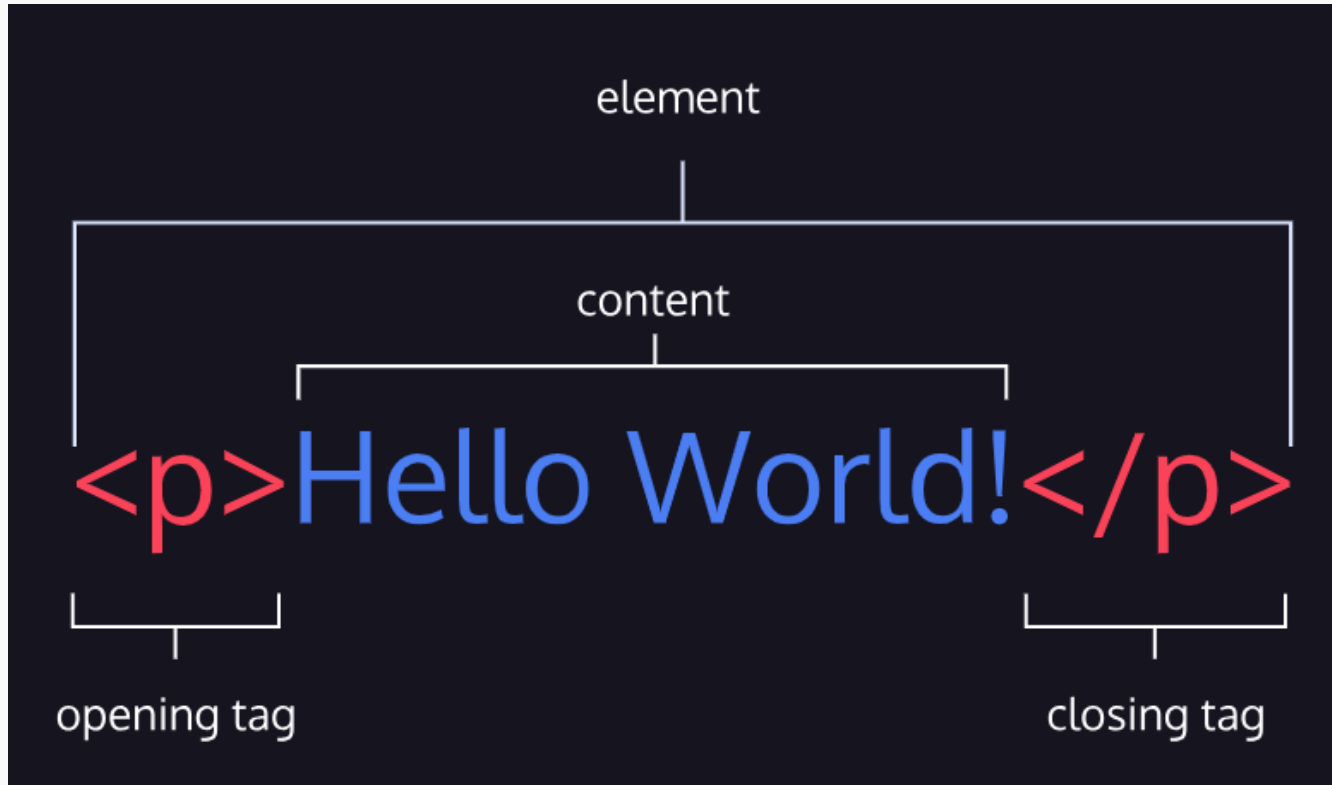
HTML (*HyperText Markup Language*) adalah bahasa **markup** standar yang digunakan untuk membuat dan menata struktur halaman web.

- bukanlah bahasa pemrograman, karena tugasnya adalah menandai dan memberi struktur pada konten website, bukan untuk melakukan logika atau perhitungan yang kompleks.
- bekerja menggunakan serangkaian *tag* untuk membungkus dan memberi arti pada berbagai jenis konten, seperti teks, gambar, video, dan elemen lainnya. Tag-tag ini kemudian diinterpretasikan oleh *browser* web untuk ditampilkan kepada pengguna dalam bentuk halaman web yang interaktif.

HTML : Fungsi

1. **Membuat Struktur Halaman Web:** HTML menyediakan elemen-elemen dasar seperti *heading* (judul), paragraf, daftar, tabel, dan lainnya untuk menyusun konten website secara logis dan terstruktur.
2. **Menandai Konten:** HTML memungkinkan kita untuk menandai berbagai jenis konten, seperti teks tebal, miring, atau garis bawah, serta menambahkan gambar, video, dan elemen multimedia lainnya.
3. **Membuat Tautan (Link):** HTML memungkinkan kita untuk membuat tautan atau *link* yang menghubungkan halaman web satu sama lain, atau ke sumber daya eksternal lainnya.
4. **Menyediakan Informasi Tambahan (Metadata):** HTML juga memungkinkan kita untuk menambahkan metadata, yaitu informasi tambahan tentang halaman web yang tidak ditampilkan secara langsung kepada pengguna, tetapi penting untuk *search engine* (SEO) dan keperluan lainnya.

HTML Anatomy



Basic Tag

[]

Contoh-contoh tag dasar:

<html>, <head>, <title>, <body>, <h1> sampai <h6>, <p>, <a>, , , , , dll.



Cheatsheet

<https://www.codecademy.com/learn/learn-html/modules/learn-html-elements/cheatsheet>

Complete Reference



[]

Contoh HTML (1)

«« <!DOCTYPE html>

<html>

<head>

<title>Tentang Kucing</title>

</head>

<body>

<h1>Kucing</h1>

<p>Kucing adalah hewan mamalia karnivora dari keluarga Felidae. Kucing telah berdampingan dengan manusia selama setidaknya 9.500 tahun, dan saat ini merupakan salah satu hewan peliharaan paling populer di dunia.</p>

»»

Contoh HTML (2)

<h2>Jenis-jenis Kucing</h2>

Anggora

Persia

Siam

Maine Coon

<h2>Gambar Kucing</h2>

</body>

</html>



Tentang Kucing



File

C:/Users/reisa/Desktop/kucing.html



Kucing

Kucing adalah hewan mamalia karnivora dari keluarga Felidae. Kucing telah berdampingan dengan manusia selama setidaknya 9.500 tahun, dan saat ini merupakan salah satu hewan peliharaan paling populer di dunia.

Jenis-jenis Kucing


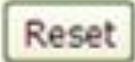
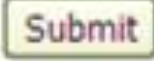

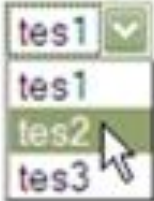
- Anggora
- Persia
- Siam
- Maine Coon

Gambar Kucing



Tag Tambahan

Code	Fungsi	Preview
<code><form></form></code>	Mengirim inputan dari user.	
<code><input type="text" /></code>	Membuat textfield	<input type="text" value="tes1"/>
<code><input type="checkbox" /></code>	Membuat checkbox	<input type="checkbox"/> tes1
<code><input type="radio" /></code>	Membuat radio button	<input type="radio"/> tes1

<code><input type="button" value="ok" /></code>	Membuat tombol	
<code><input type="reset" /></code>	Membuat tombol reset	
<code><input type="submit" /></code>	Membuat submit	
<code><input type="password" /></code>	Membuat input seperti textfield yang nilainya disamarkan	
<code><select></code> <code> <option>tes1</option></code> <code> <option>tes2</option></code> <code> </code> <code></select></code>	Membuat combobox	



02 Validasi HTML

{ }

1. Validasi Formulir

Validasi formulir adalah proses penting untuk memastikan data yang dimasukkan pengguna valid dan sesuai dengan yang diharapkan. HTML5 menyediakan beberapa atribut bawaan untuk validasi formulir:

- **required**: Atribut ini menandai bahwa suatu input harus diisi sebelum formulir dapat dikirim. []
- **pattern**: Atribut ini menentukan pola yang harus diikuti oleh input. Misalnya, kita dapat menggunakan **pattern** untuk memastikan input nomor telepon hanya berisi angka. { }
- **type**: Atribut ini menentukan jenis input, seperti **email**, **number**, atau **date**. Browser akan otomatis melakukan validasi dasar berdasarkan jenis input. >>

Coba tentukan setiap input ini jenis validasi apa?

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Formulir Validasi</title>
5 </head>
6 <body>
7
8   <h1>Formulir Pendaftaran</h1>
9
10  <form action="/submit" method="post">
11    <label for="nama">Nama Lengkap:</label><br>
12    <input type="text" id="nama" name="nama" required><br><br>
13
14    <label for="email">Email:</label><br>
15    <input type="email" id="email" name="email" required><br><br>
16
17    <label for="usia">Usia:</label><br>
18    <input type="number" id="usia" name="usia" min="18" required><br><br>
19
20    <label for="password">Password:</label><br>
21    <input type="password" id="password" name="password" minlength="8" required><br><br>
22
23    <input type="submit" value="Daftar">
24  </form>
25
26 </body>
27 </html>
```

Formulir Pendaftaran

Nama Lengkap:

Email:

Usia:

Password:

Daftar

2. Validasi Struktur Dokumen

Selain validasi formulir, penting juga untuk memastikan struktur dokumen HTML valid. Beberapa hal yang perlu diperhatikan:

- **Tag yang Ditutup:** Pastikan semua tag HTML dibuka dan ditutup dengan benar. Tag yang tidak ditutup dapat menyebabkan masalah tampilan.
- **Urutan Tag:** Perhatikan urutan tag yang benar. Misalnya, tag `<head>` harus berada di dalam tag `<html>`, dan tag `<body>` juga harus berada di dalam tag `<html>`.
- **Atribut yang Benar:** Pastikan semua atribut yang digunakan valid dan sesuai dengan tag yang bersangkutan.

{ }

Cari mana yang tidak valid ?!

[]

```
1 <html>
2 <head>
3   <title>Contoh Halaman</title>
4 </head>
5 <body>
6   <h1>Selamat Datang</h1>
7   <p>Ini adalah contoh paragraf.
8 </body>
9 </html>
```

Setelah diperbaiki

```
1 <html>
2 <head>
3   <title>Contoh Halaman</title>
4 </head>
5 <body>
6   <h1>Selamat Datang</h1>
7   <p>Ini adalah contoh paragraf.</p>
8 </body>
9 </html>
```

[] Kode di atas tidak valid karena tag `<p>` tidak ditutup dengan benar.

Mengapa Validasi Penting?

- Penting untuk memastikan website berfungsi dengan baik di berbagai browser.
- Mencegah kesalahan tampilan dan perilaku website.
- Meningkatkan SEO (Search Engine Optimization).

Cara Validasi struktur HTML

- Menggunakan tools validasi HTML online, seperti
 - W3C Markup Validation Service. <https://validator.w3.org/>
 - Nu HTML Checker: <https://validator.w3.org/nu>
- Langsung di VSCode :
 - Ekstensi HTML Validator
 - Fitur Emmet



[]

03

Intro GIT

{ }

Definisi Git

Git adalah sebuah sistem kontrol versi (*Version Control System*) yang sangat populer di kalangan pengembang perangkat lunak. Sederhananya, Git membantu Anda mencatat dan mengelola semua perubahan yang Anda lakukan pada kode program atau file proyek Anda.

Bayangkan Anda sedang menulis sebuah dokumen penting. Anda ingin menyimpan beberapa versi dokumen tersebut agar Anda bisa melihat kembali perubahan yang telah Anda lakukan atau mengembalikan dokumen ke versi sebelumnya jika diperlukan. Git melakukan hal yang sama untuk kode program Anda.

Fungsi Utama Git

« **Mencatat Perubahan** : seperti menambahkan baris kode baru, menghapus kode, atau mengubah kode yang sudah ada. (dapat melihat riwayat perubahan dari waktu ke waktu)

Mengelola Versi: Setiap versi ini disebut *commit*. dapat dengan mudah beralih antar versi atau mengembalikan proyek ke versi sebelumnya jika terjadi kesalahan. []

[] **Kolaborasi Tim**: bekerja pada proyek yang sama secara bersamaan tanpa takut terjadi konflik atau kehilangan perubahan. Git menyediakan mekanisme untuk menggabungkan perubahan . { }

Backup dan Pemulihan: Jika terjadi masalah pada komputer lokal, dapat dengan mudah mengembalikan proyek dari repositori Git. »

Istilah-Istilah Penting dalam Git

- **Repositori (*Repository*)**: Tempat penyimpanan semua file dan riwayat perubahan proyek Anda.
- **Commit**: Snapshot atau rekaman perubahan pada proyek Anda pada waktu tertentu.
- **Branch**: Cabang pengembangan yang memungkinkan Anda mengerjakan fitur baru tanpa mengganggu kode utama.
- **Merge**: Proses penggabungan perubahan dari satu branch ke branch lainnya.

<https://education.github.com/git-cheat-sheet-education.pdf>

Skenario

Tiga orang (Alice, Bob, dan Charlie) bekerja pada proyek website sederhana. Mereka akan menggunakan Git untuk berkolaborasi dan mengelola perubahan kode HTML mereka.

Langkah-langkah Git (1)

1. Membuat Repositori Git:

Salah satu anggota tim (misalnya, Alice) membuat repositori Git pusat di platform seperti GitHub, GitLab, atau Bitbucket. Alice mengunggah file HTML awal (misalnya, `index.html`) ke repositori tersebut.

2. Mengkloning Repositori:

Bob dan Charlie mengkloning repositori ke komputer lokal mereka masing-masing menggunakan perintah `git clone <URL repositori>`.

3. Membuat Branch:

Setiap anggota tim membuat branch lokal untuk mengerjakan fitur atau bagian halaman web yang berbeda. Misalnya:

- Alice membuat branch `fitur-header`
- Bob membuat branch `fitur-konten`
- Charlie membuat branch `fitur-footer`

Langkah-langkah Git (2)

4. Mengerjakan Fitur:

Setiap anggota tim bekerja pada branch mereka masing-masing dan melakukan perubahan pada file HTML yang diperlukan.

Mereka melakukan *commit* secara berkala untuk menyimpan perubahan mereka dengan perintah `git add .` (untuk menambahkan semua perubahan) dan `git commit -m "Pesan commit"`.

5. Mengirim Perubahan:

Setelah selesai dengan fitur mereka, setiap anggota tim mengirim (push) branch mereka ke repositori pusat menggunakan perintah `git push origin <nama branch>`.

6. Membuat Pull Request:

Setelah branch mereka di-*push*, setiap anggota tim membuat *pull request* di platform repositori. *Pull request* ini memberitahukan kepada anggota tim lain bahwa ada perubahan yang siap untuk ditinjau dan digabungkan (merge) ke branch utama.

Langkah-langkah Git (3)

7. Meninjau Perubahan:

Anggota tim lain meninjau perubahan yang diajukan dalam *pull request*. Mereka dapat memberikan komentar atau saran jika diperlukan.

8. Menggabungkan Perubahan:

Setelah perubahan disetujui, salah satu anggota tim (biasanya Alice sebagai pemilik repositori) menggabungkan *pull request* tersebut ke branch utama (misalnya, `main` atau `master`). []

9. Memperbarui Branch Lokal:

Setelah *pull request* digabungkan, Bob dan Charlie harus memperbarui branch lokal mereka dengan branch utama menggunakan perintah `git pull origin main`. { }

10. Mengatasi Konflik (Jika Ada):

Jika ada perubahan yang saling bertentangan (konflik) antara branch yang berbeda, anggota tim harus menyelesaikan konflik tersebut secara manual sebelum dapat digabungkan. >>



How ?

Mudah bukan ?

Thanks!



CREDITS: This presentation template was created by Slidesgo, and includes icons by Flaticon, and infographics & images by Freepik

Please keep this slide for attribution



{({({ >> })) << }



(({ >> 0 | □ □ □ }))

```
((: 00 - =>> } )  
{ (<1 00 1 000 >> )}  
((: 0)>"< )  
<01 001} +100 0}>  
((: 0)>"< )  
{ (<1 00 1 000 >> )}
```

