

Pre-Material

Bootcamp

ODP IT Fundamental : Elevating Your IT Skills to The Next Level

ODP Batch 22 & 24



Introduction to Full Stack Web Developer

No	Title	Category	Link
1	PPT - Introduction to Full Stack Web Developer	PPT	[Click Here]
2	Reading - Introduction to Full Stack Web Developer	PPT	[Click Here]
3	Opening	Video	[Click Here]
4	Pengenalan Full Stack Web Devloper	Video	[Click Here]
5	Pengenalan Jobdesc Full Stack Web Devloper	Video	[Click Here]
6	Pengenalan Skillset Full Stack Web Devloper	Video	[Click Here]
7	Pengenalan Tools Full Stack Web Devloper	Video	[Click Here]

Software Development Life Cycle

No	Title	Category	Link
1	Reading - SDLC	PPT	[Click Here]
2	Reading - SDLC	PPT	[Click Here]

Tech Stack & MERN Tech Stack

No	Title	Category	Link
1	Reading - Tech Stack	Article	[Click Here]
2	Reading - Mern Stack	Article	[Click Here]
3	Reading - Top Tech Stack	Article	[Click Here]

Introduction - Internet

No	Title	Category	Link
1	Opening	Video	[Click Here]
2	Bagaimana internet bekerja?	Video	[Click Here]
3	Pengenalan HTTP	Video	[Click Here]
4	Pengenalan Metodologi Agile	Video	[Click Here]
5	Pengenalan Bahasa Pemrograman	Video	[Click Here]

HTML

No	Title	Category	Link
1	Reading - HTML & CSS Framework	Article	[Click Here]
2	PPT - HTML	PPT	[Click Here]
3	Opening	Video	[Click Here]
4	Persiapan Tools	Video	[Click Here]
5	Introduction HTML and Web Design	Video	[Click Here]
6	Meta Data	Video	[Click Here]
7	HTML Text Formatting	Video	[Click Here]
8	Debugging HTML	Video	[Click Here]
9	Documents and structure	Video	[Click Here]
10	Hyperlinks	Video	[Click Here]

CSS & CSS Framework

No	Title	Category	Link
1	PPT - CSS & CSS Framework	PPT	[Click Here]
2	Opening	Video	[Click Here]
3	Persiapan Tools	Video	[Click Here]
4	CSS Introduction	Video	[Click Here]
5	CSS Syntax	Video	[Click Here]
6	Selector	Video	[Click Here]
7	CSS Values and Units	Video	[Click Here]
8	The Box Models	Video	[Click Here]
9	Debugging CSS	Video	[Click Here]
10	Bootstrap	Video	[Click Here]

Layouting UI

No	Title	Category	Link
1	PPT - Layouting UI	PPT	[Click Here]
2	Opening	Video	[Click Here]
3	Understanding Web Layout	Video	[Click Here]
4	Layouting technique	Video	[Click Here]
5	Slicing	Video	[Click Here]
6	Practicing CSS	Video	[Click Here]

UI Responsive

No	Title	Category	Link
1	PPT - UI Responsive	PPT	[Click Here]
2	Opening	Video	[Click Here]
3	Get to know Responsive Design	Video	[Click Here]
4	Media Queries	Video	[Click Here]
5	Unit	Video	[Click Here]
6	Responsive Design Technique	Video	[Click Here]

Introduction to Javascript

No	Title	Category	Link
1	Reading - Javascript Fundamental	PPT	[Click Here]
2	PPT - Introduction to Javascript	PPT	[Click Here]
3	Opening	Video	[Click Here]
4	Introduction to Javascript	Video	[Click Here]
5	Javascript Versions	Video	[Click Here]
6	How to run Javascript	Video	[Click Here]

Basic Javascript

No	Title	Category	Link
1	PPT - Basic Javascript	PPT	[Click Here]
2	Opening	Video	[Click Here]
3	Variables	Video	[Click Here]
4	Data Types	Video	[Click Here]
5	Data Structures	Video	[Click Here]

Basic Javascript Technique

No	Title	Category	Link
1	Reading - Basic Javascript Technique & Version Control	PPT	[Click Here]
2	Basic Javascript Technique	PPT	[Click Here]
3	Opening	Video	[Click Here]
4	Algorithm, Flowchart, Pseudocode	Video	[Click Here]
5	Loops and Iterations	Video	[Click Here]
6	Control Flow	Video	[Click Here]
7	Functions	Video	[Click Here]

Command Line

No	Title	Category	Link
1	PPT - Command Line	PPT	[Click Here]
2	Opening	Video	[Click Here]
3	Terminal and IDE	Video	[Click Here]

Operator & Expression

No	Title	Category	Link
1	PPT - Operator & Expression	PPT	[Click Here]
2	Opening	Video	[Click Here]
3	Operators	Video	[Click Here]
4	Logical Operators	Video	[Click Here]

GIT

No	Title	Category	Link
1	PPT - GIT	PPT	[Click Here]
2	Opening	Video	[Click Here]
3	Introduction to GIT	Video	[Click Here]
4	How to use GIT	Video	[Click Here]
5	Installing, initializing and committing GIT	Video	[Click Here]
6	Remote repository	Video	[Click Here]
7	Branching	Video	[Click Here]
8	Commit behind	Video	[Click Here]

Javascript DOM

No	Title	Category	Link
1	DOM Intro	Article	[Click Here]
2	DOM Methods	Article	[Click Here]
3	DOM Document	Article	[Click Here]
4	DOM Elements	Article	[Click Here]
5	DOM HTML	Article	[Click Here]
6	DOM Forms	Article	[Click Here]
7	DOM CSS	Article	[Click Here]
8	DOM Event	Article	[Click Here]
9	DOM Listeners	Article	[Click Here]

Thank You

Bootcamp

ODP IT Fundamental : Elevating Your IT Skills to The Next Level

ODP Batch 22 & 24



D-01 Training

Programming

Introduction



TABLE OF CONTENT

Day 1 - Sesi 1

- 01 History of Apps & Web
- 02 Product Concept
- 03 Software Development Lifecycle (SDLC)
- 04 Techstack

1. Sejarah Program dan Web

Melihat bagaimana dulu aplikasi dibuat



Mengapa desain database penting?

Aplikasi

Web

Komputer

Bahasa
Pemrograman

Internet

Aplikasi Desktop

Aplikasi Web

Aplikasi Mobile

Tugas Individu

Silahkan riset untuk mencari hal - hal penting apa saja yang terkait dengan kata kunci di sebelah..

Komputer

**Bahasa
Pemrograman**

Internet

Aplikasi Desktop

Aplikasi Web

Aplikasi Mobile

2. Konsep Produk

Penyelesaian masalah yang kita buat menjadi bernilai ekonomi



Konsep yang dipakai dalam membuat “produk terbaik”

Konsep Produk

Konsep Produksi

Filosofi

Konsep Produk

Konsumen menginginkan produk yang berkualitas tinggi dan mempunyai banyak fitur.

Konsep Produksi

Konsumen menginginkan produk yang tersedia di mana - mana dan harganya serendah mungkin.

Cara

Konsep Produk

Secara terus - menerus meningkatkan kualitas produk.

Konsep Produksi

Memperbesar efisiensi produksi dan kanal distribusi.

Tujuan

Konsep Produk

Menurut Kamu apa?

Konsep Produksi

Menurut Kamu apa?

Keunggulan

Konsep Produk

Menurut Kamu apa?

Konsep Produksi

Menurut Kamu apa?

Kelemahan

Konsep Produk

Menurut Kamu apa?

Konsep Produksi

Menurut Kamu apa?

Diskusi: untuk pengembangan perangkat lunak, konsep manakah yang lebih cocok dipakai?

Konsep Produk

Konsep
Produksi

Buat Product
Concept
Statement

Evaluasi
Product
Concept
Statement
terhadap
Business
Goals

Lakukan
Concept
Testing

Kembangkan
Produk yang
sesuai dengan
Product
Concept yang
disukai
Pelanggan

Gunakan
Product
Concept ke
Dalam
Marketing
Strategy

Implementasi Konsep Produk

Buat Product
Concept
Statement

Evaluasi
Product
Concept
Statement
terhadap
Business
Goals

Lakukan
Concept
Testing

Kembangkan
Produk yang
sesuai dengan
Product
Concept yang
disukai
Pelanggan

Gunakan
Product
Concept ke
Dalam
Marketing
Strategy

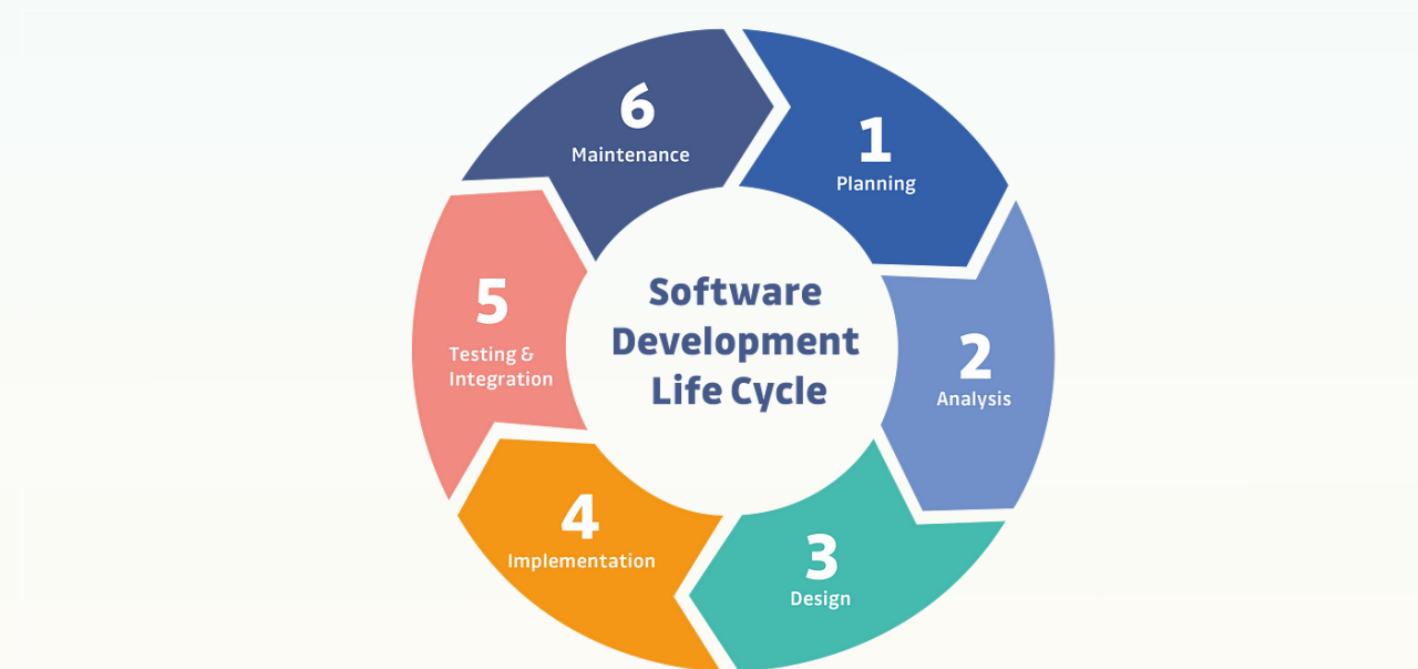
**Diskusi: di bagian manakah kemampuan
programming kamu dapat digunakan?**

3. SDLC

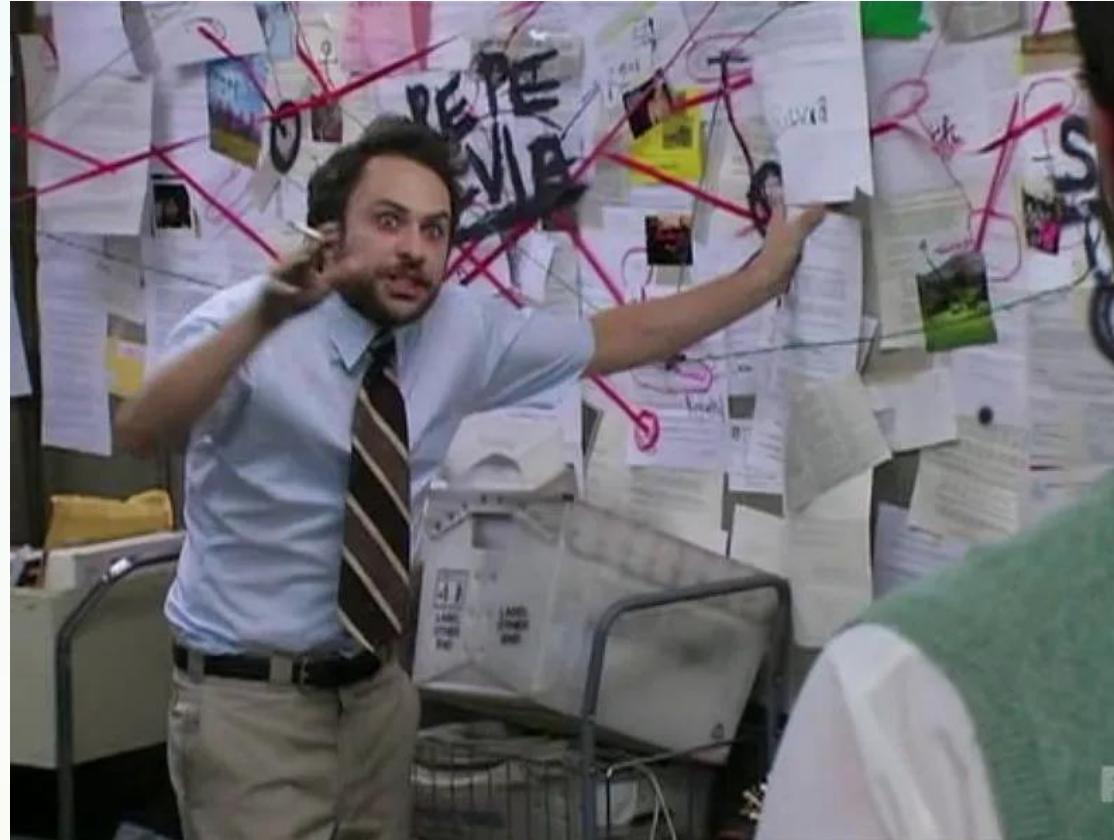
Siklus hidup pengembangan aplikasi



Software Development Life Cycle adalah metodologi yang menggunakan proses - proses tertentu dari awal hingga selesai untuk mengembangkan perangkat lunak.



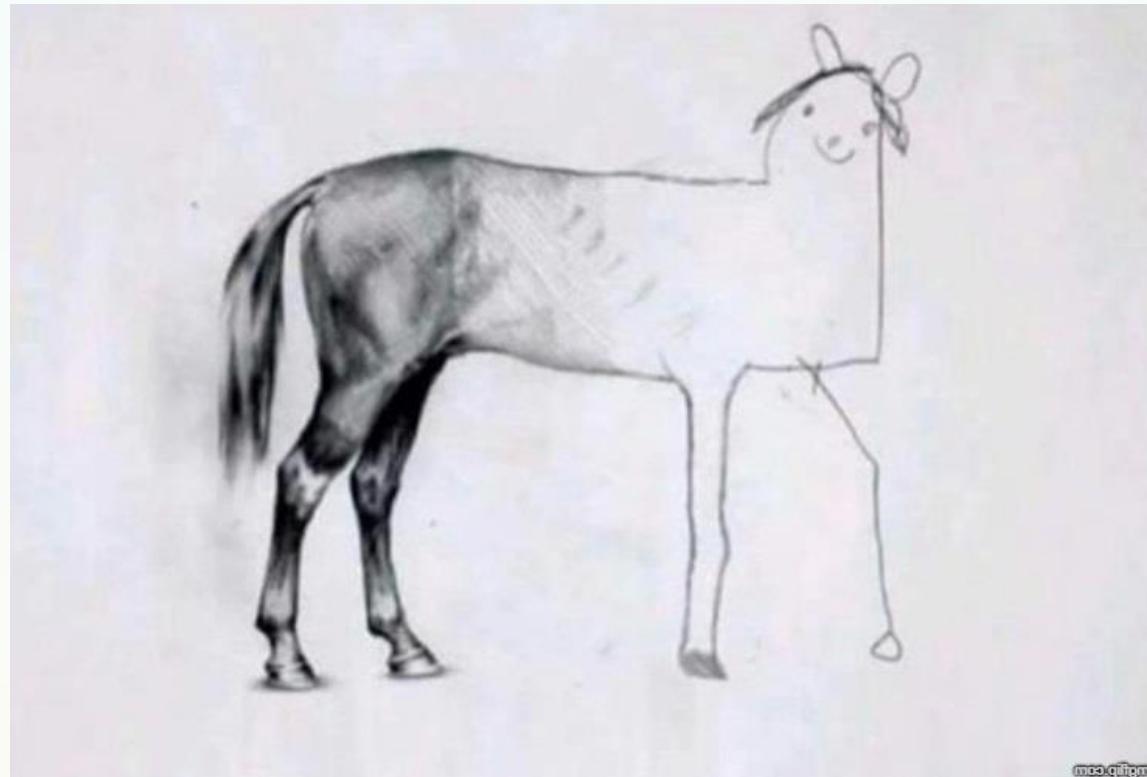
Planning



Analysis



Design



Implementation



Testing



Deployment



Pengelompokan Model SDLC

Adaptive

Membuat perangkat lunak dengan melakukan analisa dan perencanaan secukupnya dan melakukan desain, pengembangan, testing dan implementasi (jika mungkin) atas kebutuhan di atas.

Predictive

Membuat perangkat lunak dengan melakukan analisa kebutuhan dan perencanaan secara menyeluruh terlebih dahulu, lalu melakukan desain, pengembangan, testing dan implementasi sampai **semua kebutuhan** sudah dibuat.

Keunggulan

Adaptive

Lebih mungkin meningkatkan kepuasan pelanggan karena perangkat lunak sudah bisa dipakai dalam waktu singkat dan bisa berubah sesuai kebutuhan.

Predictive

Sebagian besar identifikasi dan manajemen risiko dapat dilakukan di awal.

Kelemahan

Adaptive

Perencanaan lebih minim karena hanya sedikit bagian di produk yang sudah jelas kebutuhannya.

Predictive

Sangat membutuhkan analisa kebutuhan dan perencanaan yang baik.

Contoh Model

Adaptive

Semua model berbasis Agile:
Scrum, Kanban, Extreme Programming,
Feature-Driven Development.

Predictive

Waterfall, Iterative & Incremental,
Spiral, V-Model.

Waktunya Berdiskusi!

- Manakah kelompok model yang lebih baik, adaptive atau predictive?
- Berikan contoh bidang bisnis yang pengembangan perangkat lunaknya memakai masing - masing model adaptive dan predictive.

4. Tech Stack

Teknologi yang digunakan saat membuat aplikasi



Apa itu Tech Stack?

Tech Stack adalah gabungan beberapa framework, tools dan teknologi yang kita “tumpuk” dalam membuat sebuah produk.

Diskusi:

Menurut kamu, kenapa ya kita perlu memakai tech stack tertentu?

Ternyata, tech stack menentukan tipe perangkat lunak yang dapat kita buat, level dari penyesuaian yang dapat dilakukan, dan resource yang diperlukan untuk membuat dan menjalankan perangkat lunak.

Contoh Komponen Tech Stack

Tech Stack Component	Technologies
Frontend Framework	Vue, Angular, Bootstrap
Frontend Library	React
Programming Languages	Java, C, PHP, Python, Swift, Javascript
App & Web Framework	Spring, Django
Database	MySQL, MongoDB, PostgreSQL
Event & Messaging	Kafka
Cloud Infrastructure	AWS, Azure, Google Cloud
Virtualization	Kubernetes, Docker
Operating Systems	Windows, Linux, MacOS
Environment	Node.js, JRE

Thank You

Programming

Introduction



D-01 Training

Web Layout

Structured Layout HTML and CSS



TABLE OF CONTENT

Day 1 - Sesi 4

- 01 Understanding Web Layout
- 02 Layouting Technique
- 03 Slicing
- 04 Implementing design

1. Web Layout

Web Layout



Web Layout

Tata letak situs web adalah pengaturan elemen visual pada sebuah halaman web. Ini mencakup penempatan teks, gambar, tombol, dan elemen lainnya serta bagaimana posisi mereka satu sama lain. Faktor yang dipertimbangkan dalam tata letak termasuk perataan, spasi, dan hierarki.



Key elements of web layout

Sebuah website biasanya terdiri dari beberapa element penting yang memiliki perannya masing-masing dalam design dan user experience. Berikut adalah beberapa element yang sangat penting yang membentuk suatu website:

- Navigation
- Content Area
- Footer
- Sidebar (optional)

Websites to see example of website design

- Dribbble
- Awwwards
- Behance

2. Layouting

Web Layout



Layouting Techniques

- CSS framework
- CSS float property
- CSS flexbox
- CSS grid

Layouting Techniques

- CSS framework
- CSS float property
- CSS flexbox
- CSS grid

Float

Float adalah properti CSS yang digunakan untuk mengatur elemen agar "mengapung" di sebelah elemen lain.

```
img {  
    float: left; /* Mengapung ke kiri */  
    margin-right: 10px;  
}
```

Grid

CSS Grid adalah sistem layout berbasis grid yang memungkinkan pengaturan elemen dalam dua dimensi (baris & kolom).

```
.grid-container {  
    display: grid;  
    grid-template-columns: repeat(3, 1fr);  
    gap: 10px;  
}
```

Flex vs Grid

Flexbox	Grid
Satu dimensi (horizontal atau vertikal)	Dua dimensi (baris & kolom)
Bagus untuk navbar, card layout, form	Bagus untuk dashboard, gallery, layout kompleks
Elemen dapat berubah ukuran sesuai konten	Struktur lebih terkontrol dengan baris & kolom
Gunakan justify-content & align-items untuk kontrol posisi	Gunakan grid-template-columns & grid-template-rows untuk kontrol grid
Lebih mudah dipahami	Lebih powerful, tapi butuh pemahaman lebih dalam

3. Slicing

Web Layout



Website Slicing

- Mengubah design (Figma, Adobe XD, dll) menjadi sebuah halaman web interaktif
- Memecah design ke component HTML, CSS, dan JS
- Memastikan design serta look and feel sesuai yang dinginkan



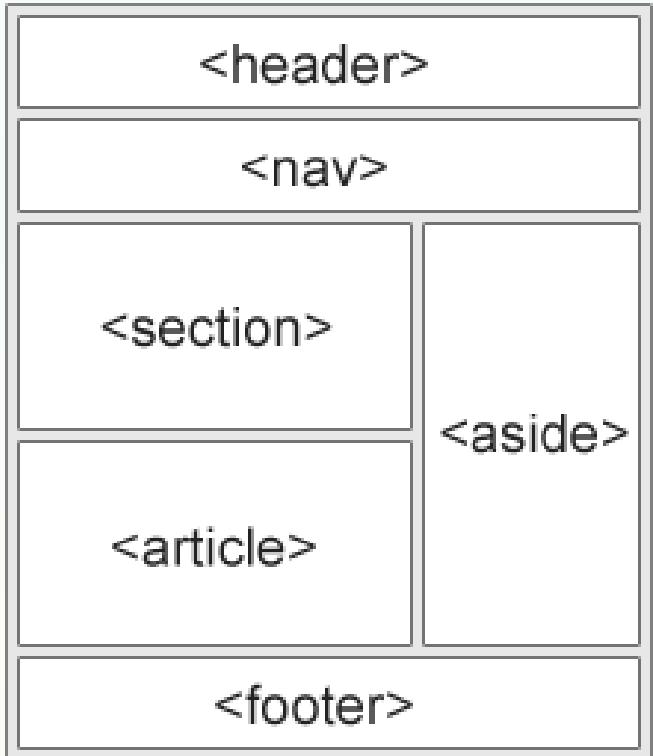
Steps to slicing

- **Analyze the Design** – Identifikasi layout, font, warna dan lain lain
- **Set Up the Project** – Tentukan apakah akan menggunakan framework atau plain
- **Create the Layout** – Gunakan **Flexbox**, **CSS Grid**, atau frameworks.
- **Apply Styles** – Implementasi bentuk font, warna, dan spasi.
- **Make it Responsive** – Menggunakan media query atau fluid layout.
- **Add Interactivity** – Memberikan interaktifitas dengan javascript.

Slicing Best Practice

- Use Semantic HTML
- Follow a Mobile-First Approach
- Optimize Images
- Use CSS Variables
- Test Across Browsers

Semantic HTML



Sebuah element semantic menjelaskan langsung makna dari element tersebut ke browser dan juga developer

4. Implementing Design

Web Layout



Hands-on

Silahkan lakukan slicing terhadap design berikut.

<https://www.figma.com/design/yYYWjLjfXmvXPggLPgBT8n/E-Wallet?node-id=0-1&t=xcmGbP1BUUOoh60E-1>

Thank You

Web Layout

Structured Layout HTML and CSS



D-01 Training

CSS

CSS Syntax & Layout



TABLE OF CONTENT

Day 1 - Sesi 3

- 01 CSS Introduction
- 02 CSS Syntax
- 03 Selector
- 04 Box Model
- 05 Flexbox

1. Pengenalan CSS

Cascading Style Sheet



CSS

CSS adalah bahasa Cascading Style Sheet dan dapat digunakan untuk mengatur tampilan elemen yang tertulis dalam bahasa markup, seperti HTML.

```
element.style {  
}  
.text-base {  
    font-size: 1rem;  
    line-height: 1.5rem;  
}  
.py-\[18px\] {  
    padding-bottom: 18px;  
    padding-top: 18px;  
}  
.px-6 {  
    padding-left: 1.5rem;  
    padding-right: 1.5rem;  
}  
.my-auto {  
    margin-bottom: auto;  
    margin-top: auto;  
}  
.mx-auto {  
    margin-left: auto;  
    margin-right: auto;  
}  
* {  
    scrollbar-color: #var(--main-surface-tertiary) transparent;  
}  
*, :after, :before {  
    --tw-border-spacing-x: 0;  
    --tw-border-spacing-y: 0;  
    --tw-translate-x: 0;  
    --tw-translate-y: 0;  
    --tw-rotate: 0;  
    --tw-skew-x: 0;  
    --tw-skew-y: 0;  
    --tw-scale-x: 1;  
    --tw-scale-y: 1;  
    --tw-pan-x: ;  
    --tw-pan-y: ;  
    --tw-pinch-zoom: ;  
    --tw-scroll-snap-strictness: proximity;  
    --tw-gradient-from-position: ;  
    --tw-gradient-via-position: ;  
    --tw-gradient-to-position: ;  
    --tw-ordinal: ;  
    --tw-slashed-zero: ;  
    --tw-numeric-figure: ;  
    --tw-numeric-spacing: ;  
}
```

FRAMEWORK CSS

Ada banyak framework CSS, pada dasarnya framework CSS ada untuk mempermudah dalam pengembangan tanpa perlu mendefinisi satu persatu desain yang ingin ditampilkan.

Beberapa framework CSS seperti:

- Bulma
- Tailwind
- Semantic UI
- Bootstrap



Before & After CSS

Heading Blue and Center

Paragraph Red.

Heading CSS Internal

Heading CSS Eksternal CSS

Heading Blue and Center

Paragraph Red.

Heading CSS Internal

Heading CSS Eksternal CSS

2. CSS Syntax

Cascading Style Sheet



Tipe CSS

Ada 3 tipe CSS, yaitu

- Inline
- Internal
- Eksternal

```
element.style {
    overflow: scroll;
}

body {
    height: auto;
    overflow-y: unset!important;
    overflow-x: hidden!important;
}

html, body {
    font-family: Nunito-Sans,-apple-system,BlinkMacSystemFont,Segoe UI,Roboto;
    scroll-behavior: smooth;
    width: 100%;
    min-height: 100%;
    margin: 0;
    padding: 0;
}

body {
    margin: 0;
    color: #000000d9;
    font-size: 14px;
    font-family: -apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Helvetica
    Neue,Arial,Noto Sans,sans-serif,"Apple Color Emoji","Segoe UI Emoji",Segoe UI
    Symbol,"Noto Color Emoji";
    font-variant: tabular-nums;
    line-height: 1.5715;
    background-color: #ffff;
    font-feature-settings: "tnum";
}

html, body {
    width: 100%;
    height: 100%;
}
```

Inline CSS

CSS jadi atribut pada sebuah element di html

```
<body>
  <h1 style="color: blue;text-align:center;">Heading Blue
  and Center</h1>
  <p style="color: red;">Paragraph Red.</p>
</body>
```

Heading Blue and Center

Paragraph Red.

Internal CSS

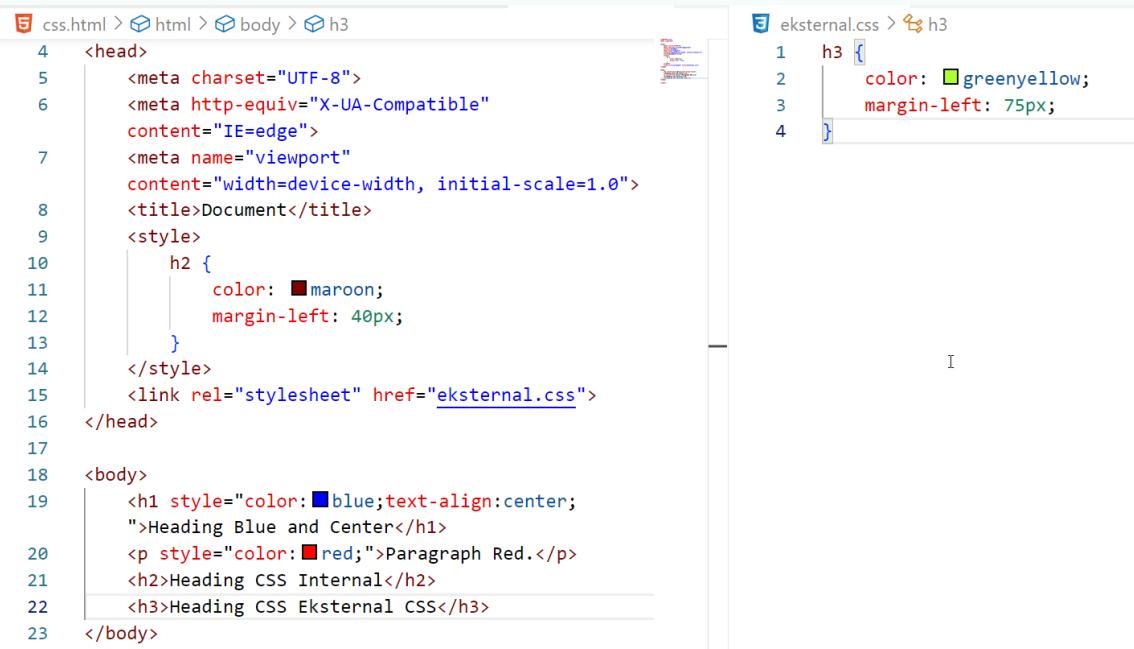
CSS internal berada pada tag head dan di dalam tag style

```
<style>
  h2 {
    color: maroon;
    margin-left: 40px;
  }
</style>
</head>

<body>
  <h1 style="color: blue;text-align:center;">Heading Blue
  and Center</h1>
  <p style="color: red;">Paragraph Red.</p>
  <h2>Heading CSS Internal</h2>
</body>
```

Eksternal CSS

CSS berada di luar file html, tujuannya dapat digunakan pada file html yang berbeda



The image shows a code editor with two tabs open. The left tab is 'css.html' and the right tab is 'eksternal.css'. The 'css.html' file contains HTML and CSS code. The 'eksternal.css' file contains a single CSS rule for the 'h3' selector.

```
css.html > html > body > h3
4  <head>
5    <meta charset="UTF-8">
6    <meta http-equiv="X-UA-Compatible"
7      content="IE=edge">
8    <meta name="viewport"
9      content="width=device-width, initial-scale=1.0">
10   <title>Document</title>
11   <style>
12     h2 {
13       color: maroon;
14       margin-left: 40px;
15     }
16   </style>
17   <link rel="stylesheet" href="eksternal.css">
18 </head>
19
20 <body>
21   <h1 style="color: blue; text-align: center;">
22     >Heading Blue and Center</h1>
23   <p style="color: red;">Paragraph Red.</p>
24   <h2>Heading CSS Internal</h2>
25   <h3>Heading CSS Eksternal CSS</h3>
26 </body>
```

```
eksternal.css > h3
1  h3 {
2    color: greenyellow;
3    margin-left: 75px;
4 }
```

Heading CSS Eksternal CSS

Value and Unit

CSS memiliki beberapa opsi satuan untuk menentukan ukuran dari berbagai propertinya. Penggunaan yang tepat dapat membuat tampilan webs akan sangat enak untuk dilihat dimana saja. Ada 2 tipe yaitu:

- **Absolute** : akan selalu memiliki ukuran yang sama apapun elemen parent-nya atau ukuran layar. Artinya properti yang diatur dengan absolute unit akan memiliki ukuran sama baik di telepon maupun di layar monitor yang besar.
- **Relative** : Relative unit berguna untuk mendesain website yang responsif karena ukurannya bisa berubah relatif terhadap parent atau ukuran layar.

Contoh Absolute

Menggunakan PX pada satuan merupakan cara agar tampilan akan menjadi absolute atau sama persis walau berganti ukuran layar

```
h1 {  
    font-size: 60px;  
}  
  
p {  
    font-size: 25px;  
    line-height: 50px;  
}
```

Contoh Relative

- % : Ukurannya relatif terhadap parent element
- em : Ukurannya relatif terhadap font-size dari elemen saat ini
- rem: Ukurannya relatif terhadap font-size root elemen (`<html>`). "rem" = "root em"
- ch: Ukurannya mengikuti jumlah karakter (1 karakter sama dengan lebar dari karakter 0/nol font yang sedang aktif)
- vh: Ukurannya relatif terhadap tinggi viewport (ukuran jendela tau aplikasi), $1vh = 1/100$ dari tinggi viewport
- vw: Ukurannya relatif terhadap lebar dari viewport. $1vw = 1/100$ lebar viewport
- vmin: Ukurannya relatif terhadap ukuran viewport yang lebih kecil (misalnya diorientasi portrait, lebar akan lebih kecil daripada tinggi). $1vmin = 1/100$ dari ukuran viewport yang lebih kecil.
- vmax: Sama dengan vmin, dia akan melihat ukuran viewport yang lebih besar
- ex: Ukurannya relatif terhadap tinggi dari karakter "x" kecil font yang sedang aktif.

3. CSS Selector

Cascading Style Sheet



CSS Selector

CSS selector adalah salah satu rangkaian aturan dari CSS yang memiliki fungsi untuk memilih suatu elemen atau lebih untuk diberikan style. Ada banyak selector yang dapat digunakan ([cek link berikut](#)) tapi dalam pembahasan ini akan dibahas beberapa selector yang sering digunakan:

- TAG
- CLASS
- ID
- Pseudo CLASS

Selector Tag

Selector ini akan memberikan style berdasarkan tag tertentu.

```
<style>
  b, i{
    color: blue;
  }
</style>
</head>
<body>
  <h1>Ini judul</h1>
  <p>Paragraph 1</p>
  <p>Paragraph 2 dan ini <b>biru bold</b></p>
  <p>Paragraph 3 dan ini <i>biru miring</i></p>
```

Ini judul

Paragraph 1

Paragraph 2 dan ini **biru bold**

Paragraph 3 dan ini *biru miring*

Selector Class

Memberikan style tanpa harus menggunakan tag karena dapat mengenai tag lainnya, maka dapat menggunakan class untuk case tersebut.

```
.backgroundgray{  
    background-color: gray;  
}  
  
</style>  
</head>  
<body>  
    <h1>Ini judul</h1>  
    <p>Paragraph 1</p>  
    <p class="backgroundgray">Paragraph 2 dan ini <b>biru bold</b></p>  
    <p>Paragraph 3 dan ini <i>biru miring</i></p>
```

Ini judul

Paragraph 1

Paragraph 2 dan ini **biru bold**

Paragraph 3 dan ini *biru miring*

Selector Class

Memberikan style tanpa harus menggunakan tag karena dapat mengenai tag lainnya, maka dapat menggunakan class untuk case tersebut.

```
.backgroundgray{  
    background-color: gray;  
}  
  
</style>  
</head>  
<body>  
    <h1>Ini judul</h1>  
    <p>Paragraph 1</p>  
    <p class="backgroundgray">Paragraph 2 dan ini <b>biru bold</b></p>  
    <p>Paragraph 3 dan ini <i>biru miring</i></p>
```

Ini judul

Paragraph 1

Paragraph 2 dan ini **biru bold**

Paragraph 3 dan ini *biru miring*

Selector ID

Sama seperti class, ID memiliki sifat yang sama

```
        }
    #backgroundpink{
        background-color: pink;
    }
}
</style>
</head>
<body>
    <h1>Ini judul</h1>
    <p>Paragraph 1</p>
    <p class="backgroundgray">Paragraph 2 dan ini <b>biru bold</b></p>
    <p id="backgroundpink">Paragraph 3 dan ini <i>biru miring</i></p>
```

Ini judul

Paragraph 1

Paragraph 2 dan ini **biru bold**

Paragraph 3 dan ini *biru miring*

Pseudo CLASS

CSS yang dapat memberikan aksi ketika disentuh atau hal lainnya.

```
.backgroundblue:hover{  
    background-color: blue;  
}  
  
</style>  
</head>  
<body>  
    <h1>Ini judul</h1>  
    <p class="backgroundblue">Paragraph 1</p>  
    <p class="backgroundgray">Paragraph 2 dan ini <b>biru bold</b></p>  
    <p id="backgroundpink">Paragraph 3 dan ini <i>biru miring</i></p>
```

Ini judul

Paragraph 1

Paragraph 2 dan ini **biru bold**

Paragraph 3 dan ini *biru miring*

4. Box Model

Cascading Style Sheet



Box Models

CSS Box Model adalah sebuah konsep dimana setiap elemen yang terdapat pada halaman web diproses sebagai kotak (box). Mulai dari paragraf, header, form, gambar, logo hingga video, sebenarnya ditampilkan oleh web browser sebagai 'box'. Pada penerapannya ada hal yang harus diketahui perbedaan antara tag yang bersifat inline dan block, dimana penerapan box model akan tepat jika pada posisi display block bukan inline.

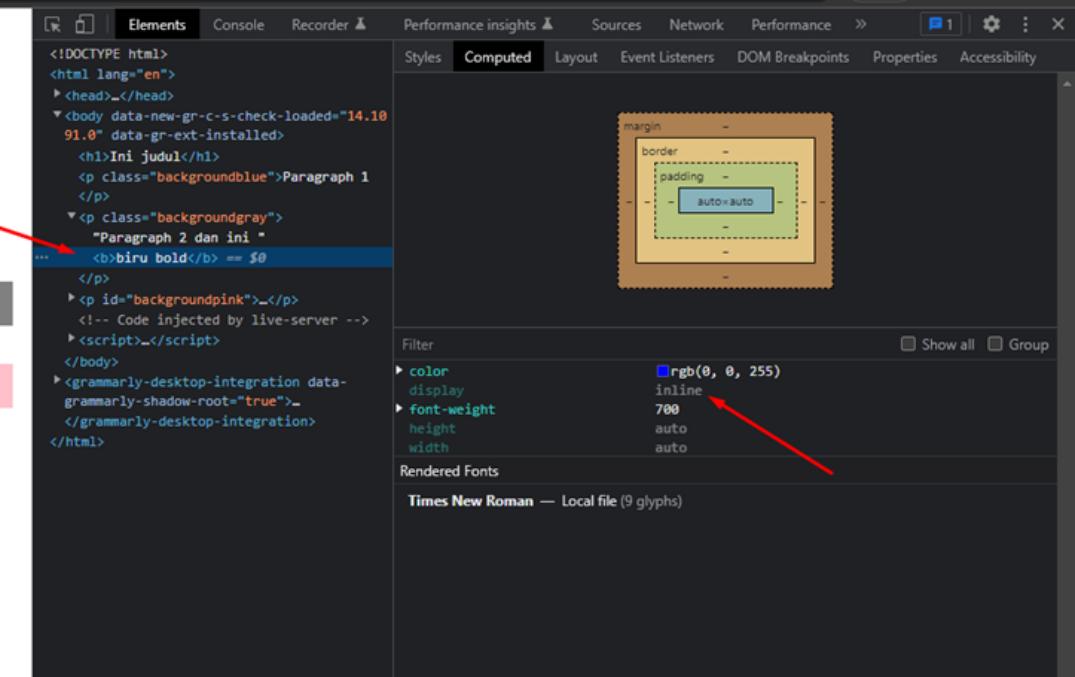
Sifat Inline

Ini judul

Paragraph 1

Paragraph 2 dan ini **biru bold**

Paragraph 3 dan ini *biru miring*



The screenshot shows the Chrome DevTools Elements tab with the Computed tab selected. A red arrow points from the text "biru bold" in the third paragraph to the corresponding CSS rule in the computed styles panel. Another red arrow points from the text "biru miring" in the third paragraph to its corresponding CSS rule. The computed styles panel displays the following properties for the selected text:

Property	Value
color	rgb(0, 0, 255)
display	inline
font-weight	700
height	auto
width	auto

The rendered font listed is Times New Roman — Local file (9 glyphs).

```
<!DOCTYPE html>
<html lang="en">
  <head>...</head>
  <body data-new-gr-c-s-check-loaded="14.1091.0" data-gr-ext-installed>
    <h1>Ini judul</h1>
    <p class="backgroundblue">Paragraph 1</p>
    <p class="backgroundgray">Paragraph 2 dan ini <b>biru bold</b> == $0</p>
    <p id="backgroundpink">...</p>
    <!-- Code injected by live-server -->
    <script>...</script>
  </body>
  <p><grammarly-desktop-integration data-grammarly-shadow-root="true">...</grammarly-desktop-integration></p>
</html>
```

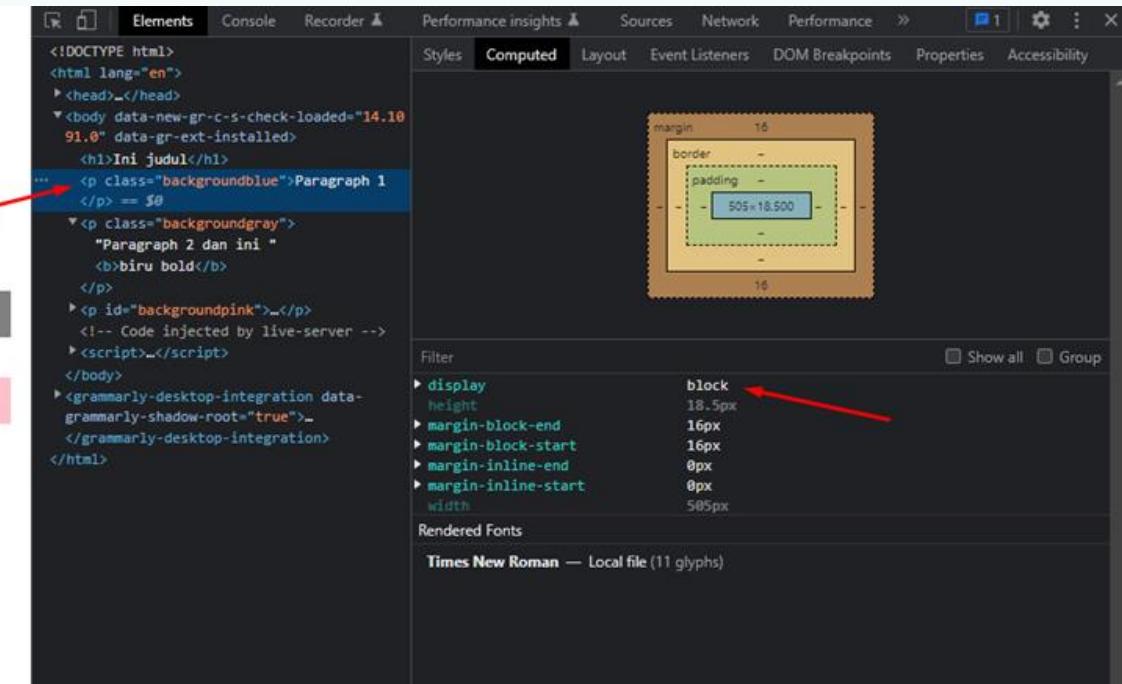
Sifat Block

Ini judul

Paragraph 1

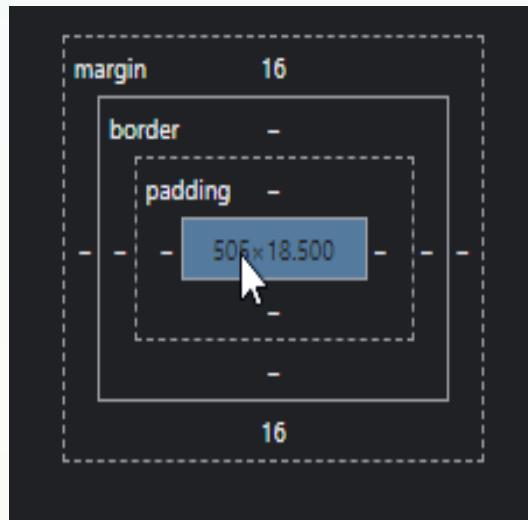
Paragraph 2 dan ini **biru bold**

Paragraph 3 dan ini *biru miring*



Element, Padding, Border, Margin

- Element adalah posisi tag itu berada
- Padding adalah jarak antara element dengan border
- Border adalah pemisah antara padding dengan margin
- Margin adalah jarak antara elemen yang satu dengan lainnya



Lihat Kelas

atau

Temukan karier yang cocok untuk saya

5. Flexbox

Cascading Style Sheet



Hands-on

Tambahkan CSS ke desain CV yang sudah kalian buat di sesi HTML sebelumnya!

Apa itu Flexbox

CSS Flexible Box layout atau sering disebut dengan Flexbox layout adalah sebuah sistem CSS yang berfungsi untuk menyederhanakan tata letak tampilan website.

Flexbox dapat mengatur elemen secara responsif baik dalam arah horizontal maupun vertikal, tanpa perlu menggunakan tata letak tradisional yang lebih kompleks seperti float atau positioning.

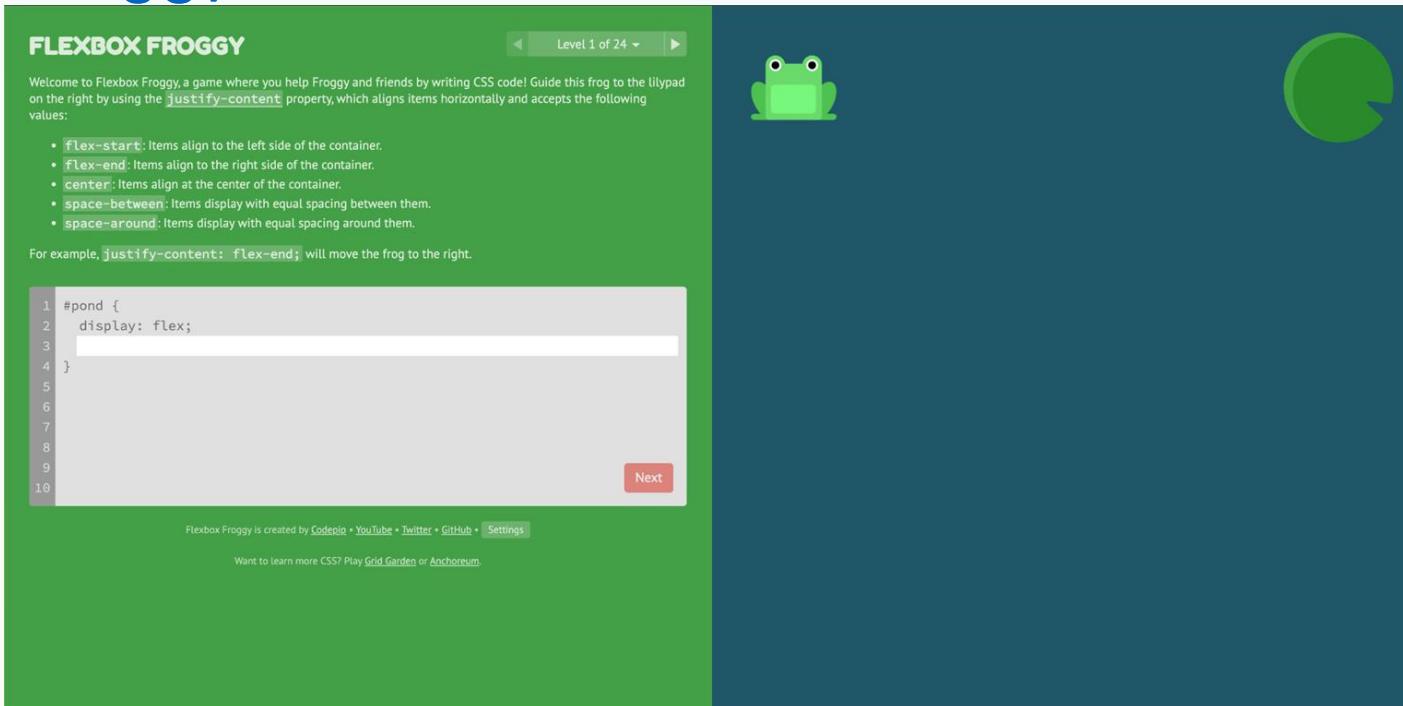
Keunggulan flexbox

- Mengatur elemen secara horizontal atau vertikal: Flexbox dapat diatur dalam satu arah, baik itu secara horizontal (row) maupun secara vertikal (column).
- Membuat elemen responsif: Elemen menggunakan layout flexbox dapat menyesuaikan ukurannya secara otomatis sesuai dengan ukuran layar sehingga dapat membuatnya lebih responsif.
- Membagi elemen dengan mudah: Properti seperti justify-content, align-items, dan align-content dapat disesuaikan dengan posisi elemen seperti tengah, kiri, kanan, atas, bawah, atau secara merata dalam kontainer.
- Membagi ruang secara proporsional: Properti seperti flex-grow dan flex-shrink, layout flexbox dapat membagi ukuran secara merata.
- Mengatur urutan elemen: Properti order dapat mengubah urutan tampilan elemen tanpa harus memindahkan elemen tersebut.
- Mengatur elemen dengan ukuran dinamis: CSS layout flexbox dapat mengelola elemen dengan ukuran berbeda di dalam kontainer yang sama.

Latihan flexbox

Untuk latihan kalian bisa mencoba mengakses:

<https://flexboxfroggy.com/>



Thank You

CSS

CSS Syntax & Layout



D-02 Training

Javascript Fundamental



TABLE OF CONTENT

Day 1 - Sesi 1

- 01 Javascript Basic
- 02 Code Structure & Variables
- 03 Data Type
- 04 Operator
- 05 Logical Operator

0. Instalasi

Javascript



Instalasi Node JS

Sebelum memulai mari kita install **Node JS** untuk bisa menjalankan javascript melalui code editor.

1. Kunjungi <https://nodejs.org>
2. Unduh versi LTS
3. Jalankan installer dan ikuti petunjuk
4. Verifikasi instalasi dengan `node -v` di Command Prompt / Terminal

1. Dasar Javascript

Javascript



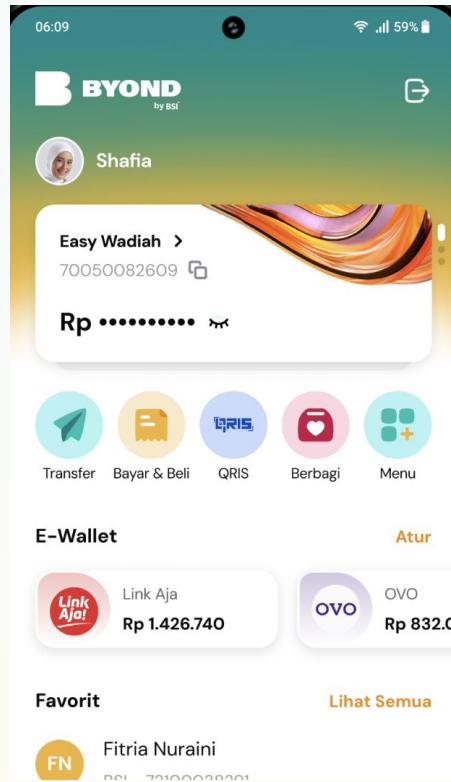
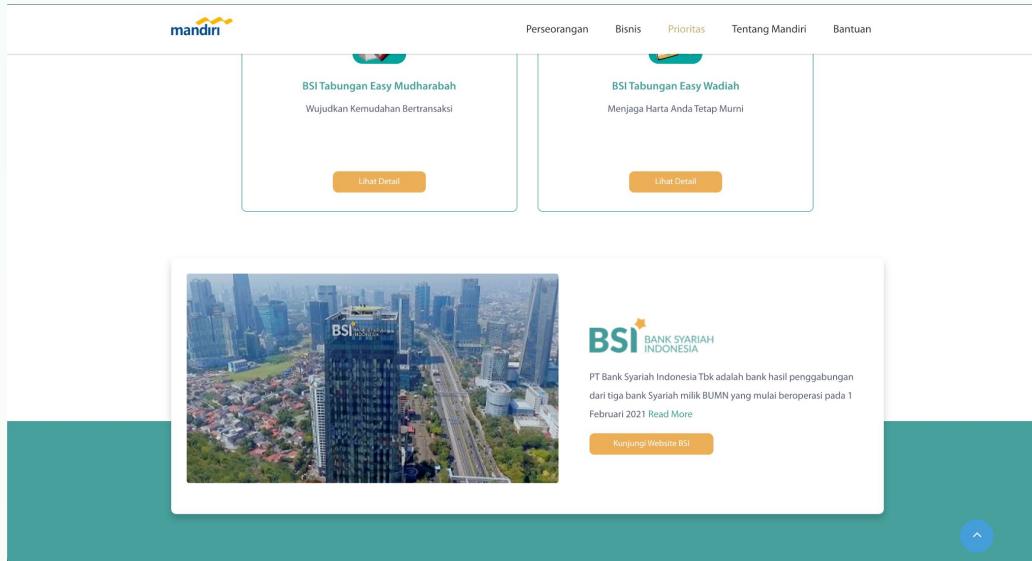
Sejarah Javascript

JavaScript dikembangkan oleh **Brendan Eich** pada tahun 1995 saat bekerja di Netscape.

- Awalnya bernama Mocha, lalu LiveScript, sebelum akhirnya dinamai JavaScript.
- Standarisasi dilakukan oleh ECMAScript (ES) sejak 1997.
- Saat ini digunakan dalam pengembangan web, backend, mobile, hingga AI.



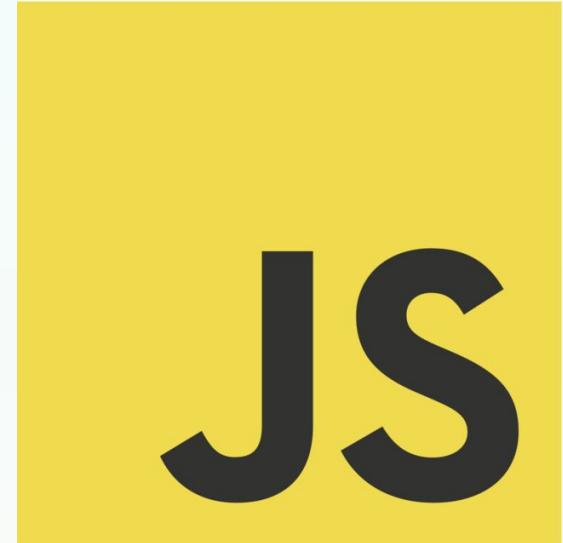
Penggunaan Javascript



Apa itu Javascript?

JavaScript adalah bahasa pemrograman yang digunakan untuk membuat halaman web interaktif.

Dapat berjalan di browser dengan bantuan V8 Engine dan Spider Money dan jalan di komputer dengan server (Node.js).



Kenapa Harus Belajar JavaScript?

2. Struktur Kode dan Variables

Javascript



Struktur Kode dalam JavaScript

Javascript biasanya terdiri dari hal berikut:

- Statement (Pernyataan)
- Expression (Ekspresi)
- Block (Blok Kode)
- Comment (Komentar)

Statement dalam javascript

Instruksi yang di eksekusi oleh javascript

```
let name = "Alice"; // Deklarasi variabel adalah statement
console.log("Hello, " + name); // Pemanggilan fungsi juga statement
```

Ekspresi dalam javascript

Ekspresi adalah kode yang mengevaluasi suatu nilai

```
let sum = 10 + 5; // 10 + 5 adalah ekspresi, hasilnya 15
let isEven = sum % 2 === 0; // ekspresi boolean, menghasilkan true atau false
console.log(isEven);
```

Block dalam javascript

Blok kode dikelompokkan menggunakan {} dan sering digunakan dalam fungsi, perulangan, dan kondisi

```
{  
  let message = "Inside block";  
  console.log(message);  
} // Variabel message hanya bisa diakses di dalam blok ini
```

Comment dalam javascript

Komentar yang diberikan untuk menjelaskan suatu kode

```
// Ini adalah komentar satu baris
let x = 5; // Variabel x diinisialisasi dengan nilai 5

/*
  Ini adalah komentar multi-baris.
  Biasanya digunakan untuk dokumentasi lebih panjang.
*/
console.log(x);
```

Variable

JavaScript menggunakan var, let, dan const untuk mendeklarasikan variabel.

- **var**: global/function scope
- **let**: block scope
- **const**: nilai tetap

Perbedaan var, let, const

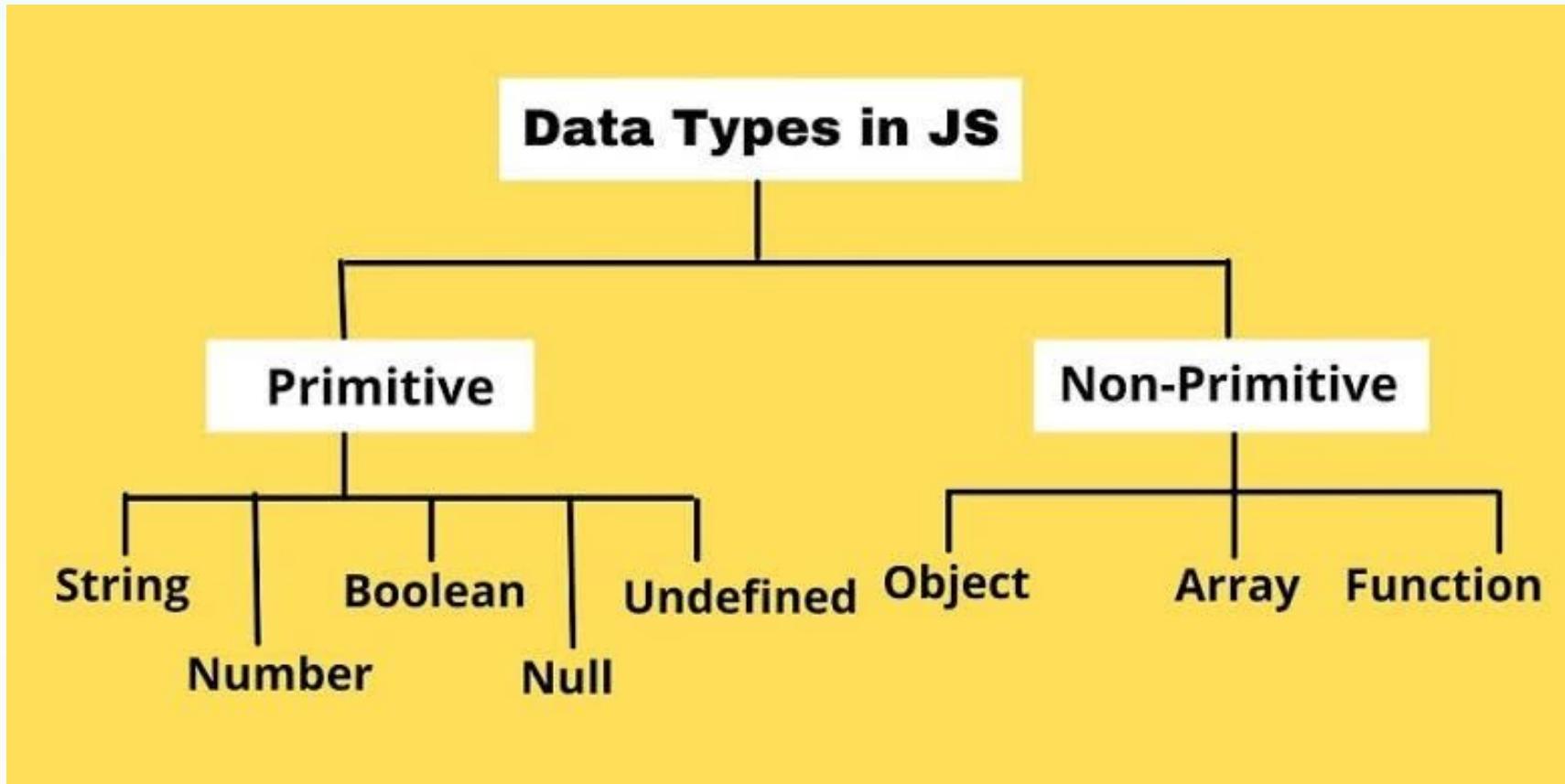
keyword	const	let	var
global scope	NO	NO	YES
function scope	YES	YES	YES
block scope	YES	YES	NO
can be reassigned	NO	YES	YES

3. Tipe data

Javascript



Tipe data di Javascript



Tipe data di Javascript

Type	typeof return value	Object wrapper
<u>Null</u>	"object"	N/A
<u>Undefined</u>	"undefined"	N/A
<u>Boolean</u>	"boolean"	<u>Boolean</u>
<u>Number</u>	"number"	<u>Number</u>
<u>BigInt</u>	"bigint"	<u>BigInt</u>
<u>String</u>	"string"	<u>String</u>
<u>Symbol</u>	"symbol"	<u>Symbol</u>

4. Operator Matematika

Javascript



Operator Aritmatika

Operator	Name	Purpose	Example
+	Addition	Menambahkan 2 angka	6 + 9
-	Subtraction	Mengurangi 2 angka	20 - 15
*	Multiplication	Mengalikan 2 angka	3 * 7
/	Division	Membagi angka sebelah kiri dengan sebelah kanan	10 / 5
%	Remainder (sometimes called modulo)	Mengembalikan hasil bagi	8 % 3 (returns 2, as three goes into 8 twice, leaving 2 left over).
**	Exponent	Raises a base number to the exponent power, that is, the base number multiplied by itself, exponent times.	5 ** 2 (returns 25, which is the same as 5 * 5).

5. Operator Logika

Javascript



Operator Logika

Operator	Name	Purpose	Example
<code>==</code>	Strict equality	Tests whether the left and right values are identical to one another	<code>5 == 2 + 4</code>
<code>!=</code>	Strict-non-equality	Tests whether the left and right values are not identical to one another	<code>5 != 2 + 3</code>
<code><</code>	Less than	Tests whether the left value is smaller than the right one.	<code>10 < 6</code>
<code>></code>	Greater than	Tests whether the left value is greater than the right one.	<code>10 > 20</code>
<code><=</code>	Less than or equal to	Tests whether the left value is smaller than or equal to the right one.	<code>3 <= 2</code>
<code>>=</code>	Greater than or equal to	Tests whether the left value is greater than or equal to the right one.	<code>5 >= 4</code>

Thank You

Javascript Fundamental



D-02 Training

Version Control

Git



1. Git

Version Control



Memahami Version Control Git

Kontrol versi adalah metode yang digunakan untuk **melacak dan mengelola perubahan** dalam kode sumber atau berkas proyek.

Git merupakan salah satu sistem kontrol versi terdistribusi yang paling populer dan kuat. Berikut adalah langkah-langkah untuk memahami kontrol versi dan Git

● **Sistem Kontrol Versi Terpusat (Centralized Version Control System)**

Dalam sistem kontrol versi terpusat, ada satu repositori sentral yang berfungsi sebagai "**master**" untuk menyimpan seluruh sejarah proyek. Setiap pengembang melakukan perubahan pada salinan lokal, kemudian mengirimkan perubahan tersebut ke repositori sentral. Contoh sistem kontrol versi terpusat adalah **Subversion (SVN)**.

● **Sistem Kontrol Versi Terdistribusi (Distributed Version Control System)**

Dalam sistem kontrol versi terdistribusi, setiap anggota tim memiliki salinan lengkap dari seluruh repositori. Ini berarti setiap pengembang memiliki salinan lengkap sejarah perubahan, tidak hanya salinan terbaru. Contoh sistem kontrol versi terdistribusi adalah **Git, Mercurial, dan Bazaar**.

Dasar Dasar Command GIT

01 git init

Menginisialisasi direktori sebagai repositori Git kosong

```
git init
```

02 git clone

Menduplikasi repositori Git yang sudah ada ke direktori lokal.

```
git clone https://github.com/username/repo.git
```

03 git status

Menampilkan status perubahan yang belum dikomit di repositori lokal.

```
git status
```

04 git add

Menambahkan perubahan ke area persiapan (staging area) untuk disiapkan menjadi commit.

```
git add file1.txt  
git add .
```

Continue...

05 git commit

Membuat commit dari perubahan yang sudah di-staging dan menambahkan pesan commit.

```
git commit -m "Menambahkan fitur login"
```

06 git push

Mengirimkan commit ke repositori jarak jauh (remote repository).

```
git push origin main
```

07 git pull

Mengambil commit terbaru dari repositori jarak jauh dan menggabungkannya ke repositori lokal.

```
git pull origin main
```

08 git branch

Menampilkan daftar cabang (branch) yang ada di repositori dan menunjukkan cabang aktif.

```
git branch
```

Continue...

09 git checkout

Beralih ke cabang lain atau ke commit tertentu.

```
git checkout feature-branch  
git checkout abc1234 (commit hash)
```

10 git merge

Menggabungkan perubahan dari satu cabang ke cabang aktif.

```
git merge feature-branch
```

11 git log

Menampilkan daftar commit beserta riwayatnya dalam repositori.

```
git log
```

12 git remote

Menampilkan daftar repositori jarak jauh yang terhubung dengan repositori lokal.

```
git remote -v
```

13 git fetch

Mengambil informasi terbaru dari repositori jarak jauh tanpa menggabungkan perubahan.

```
git fetch origin
```

Continue...

14 git diff

Menampilkan perbedaan antara versi yang sudah di-staging dengan versi sebelumnya.

```
git diff
```

15 git reset

Mengembalikan file yang sudah di-staging ke direktori kerja sebelumnya.

```
git reset file.txt
```

Simulasi Kolaborasi Menggunakan GIT

Simulasi kolaborasi Git dengan **anggota kelompok** menggunakan GitHub melibatkan **repository, pembuatan branch, membuat perubahan, mengelola konflik**, dan menyelesaiannya dengan **permintaan tarik (pull request) dan penggabungan (merge)**.

Simulasi Kolaborasi Menggunakan GIT

Silahkan clone repository berikut:
`repo_url`

Buat branch sesuai dengan nama kelompok dengan format eg. **kelompok_1**

Thank You

Version Control

Git



D-02 Training

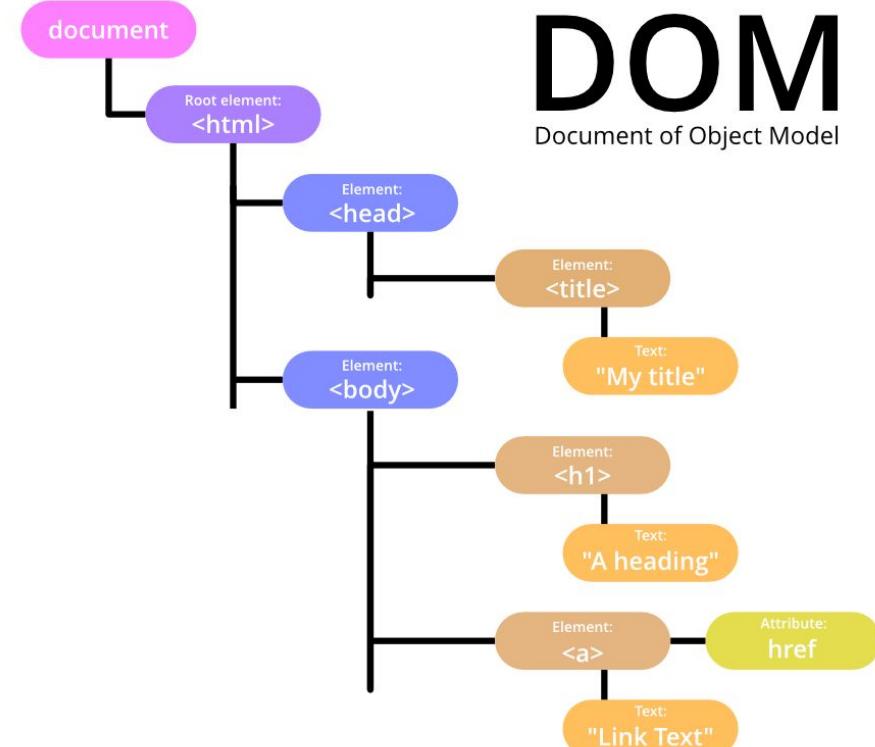
JS DOM

Document Object Model



DOM

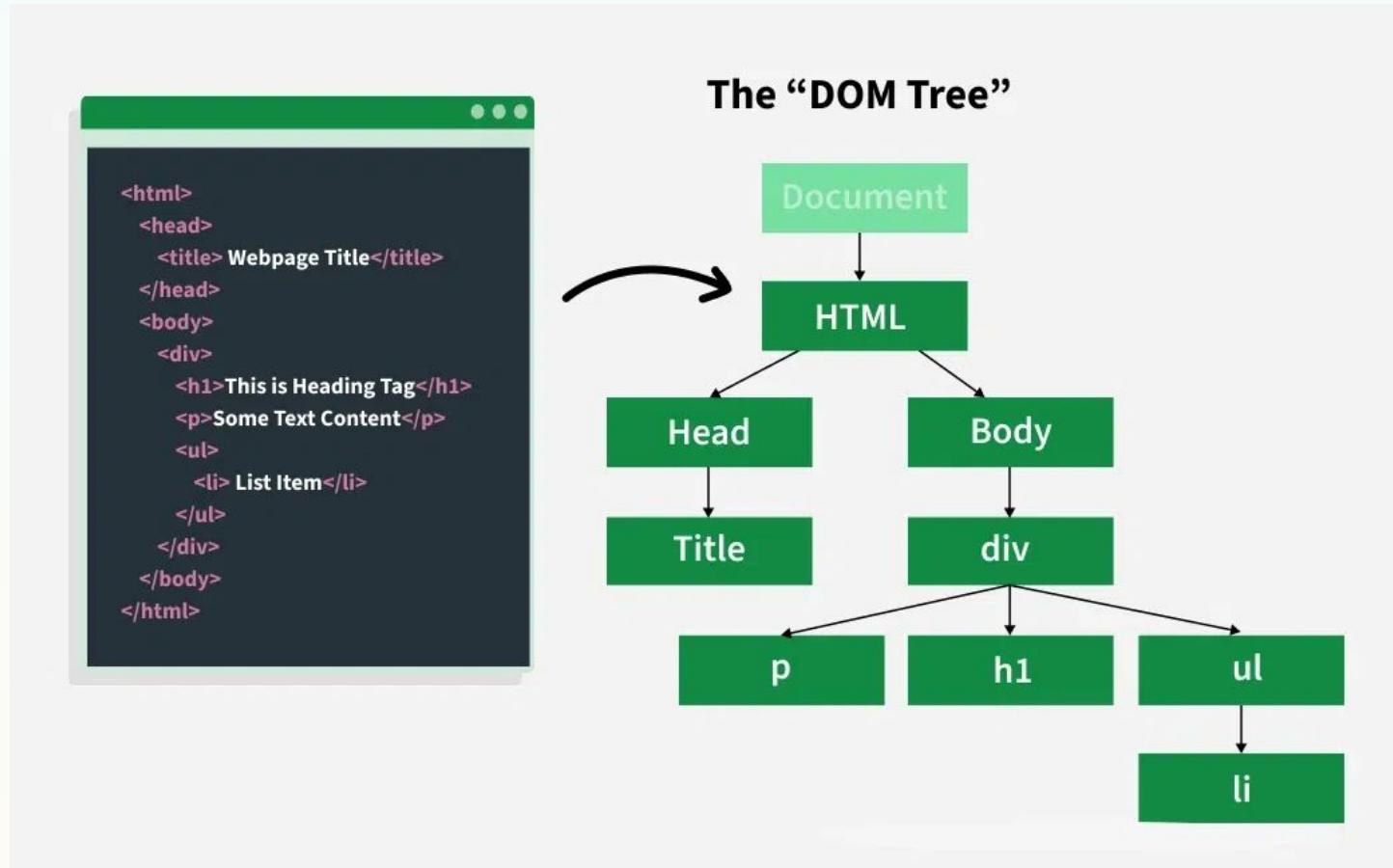
- DOM (Document Object Model) adalah representasi struktur dokumen HTML dalam bentuk pohon objek yang dapat dimanipulasi menggunakan JavaScript.
- Memungkinkan interaksi dinamis dengan halaman web.



DOM

Document of Object Model

DOM Tree



Kenapa kita butuh DOM?

- Dynamic Content Updates
- User Interaction
- Flexibility
- Cross-Platform Compatibility

Node pada DOM

- Elemen (Element Node): **<div>, <p>, <h1>**, dll.
- Atribut (Attribute Node): **id, class, src**, dll.
- Teks (Text Node): **Isi teks** dalam elemen.
- Komentar (Comment Node): **<!-- Ini komentar -->**
- Dokumen (Document Node): **Akar dari seluruh DOM.**

Mengubah konten di dalam DOM

```
<html>
  <body>
    <p id="demo"></p>

    <script>
      document.getElementById("demo").innerHTML = "Hello World!";
    </script>
  </body>
</html>
```

Menagkses element di dalam DOM

- `document.getElementById("myId");`
- `document.getElementsByClassName("myClass");`
- `document.getElementsByTagName("p");`
- **`document.querySelector(".myClass");`**
- `document.querySelectorAll("p");`

Menagkses element di dalam DOM

- **getElementById**: Cepat tetapi hanya satu elemen.
- **getElementsByClassName & getElementsByTagName**: Mengembalikan koleksi.
- **querySelector**: Fleksibel tetapi lebih lambat dibanding getElementById.
- **querySelectorAll**: Mengembalikan NodeList yang bisa di-loop dengan forEach.

Menagkses element di dalam DOM

- let parent = document.getElementById("child").parentNode;
- let firstChild = document.getElementById("parent").firstChild;
- let lastChild = document.getElementById("parent").lastChild;
- let nextSibling = document.getElementById("child").nextSibling;
- let prevSibling = document.getElementById("child").previousSibling;

Mengubah Konten dan Atribut

Mengubah teks dalam element:

```
document.getElementById("demo").textContent = "Hello World!";
```

Mengubah atribut dalam element:

```
document.getElementById("img").setAttribute("src", "image.jpg");
```

Menambahkan element

```
let newDiv = document.createElement("div");
newDiv.textContent = "Elemen baru!";
document.body.appendChild(newDiv);
```

Menghapus element

```
let parent = document.getElementById("container");
let child = document.getElementById("item");
parent.removeChild(child);
```

DOM Event

Kode javascript biasanya akan dieksekusi saat sebuah event terjadi, seperti saat user menekan suatu element HTML.

Untuk mengeksekusinya, silahkan tambahkan kode javascript pada suatu event tertentu seperti di bawah ini:

onclick=JavaScript

DOM Event Example

- Saat user click mouse
- Saat halaman web di load
- Saat suatu gambar di load
- Saat kursor mengarah ke suatu element
- Saat input field berubah
- Saat form di submit
- Saat user menekan suatu tombol di keyboard

Event Bubbling

- Ketika suatu event terjadi pada elemen anak, event tersebut akan naik ke elemen induknya sampai mencapai document (melewati semua ancestor).
- Event akan diproses dari elemen terdalam ke elemen luar

```
<div id="parent" style="padding: 20px; background: lightblue;">
    <button id="child">Klik Saya</button>
</div>

<script>
    document.getElementById("parent").addEventListener("click", function() {
        alert("Event pada Parent!");
    });

    document.getElementById("child").addEventListener("click", function() {
        alert("Event pada Child!");
    });
</script>
```

Mari latihan

- Buatlah sebuah aplikasi todo app dengan fitur sebagai berikut:

 Menambahkan tugas

 Menghapus tugas

 Menandai tugas sebagai selesai

Silahkan akses templatanya melalui: <https://pastebin.com/EKkSqNC4>

Thank You

JS DOM

Document Object Model



D-03 Training

React.js

Introduction



TABLE OF CONTENT

Day 3 - Sesi 1

- 01 What is React.js & JSX
- 02 Props, Children & Conditional Rendering
- 03 State & Render
- 04 Event & Navigation

1. Kenalan Dengan React.js

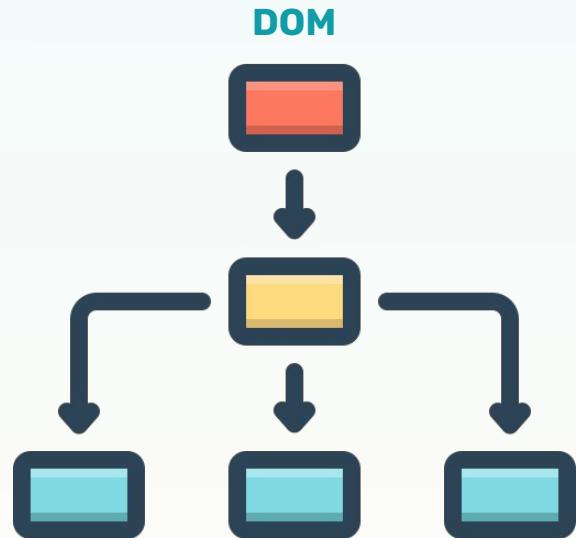
Tak kenal maka kenalan



Hal yang Harus Diketahui



Cara Website Memproses HTML



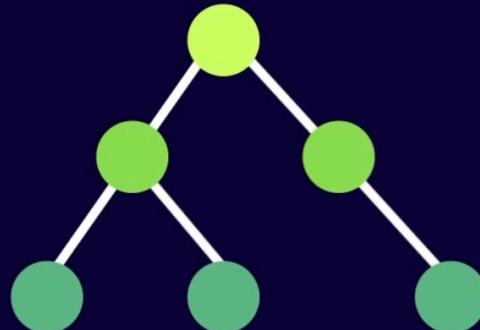
Vanilla JavaScript

```
const btn = document.querySelector('#btn');
btn.addEventListener('click', () => {
  const div = document.querySelector('#div');
  div.style.display = 'none';
});
```

COMPONENTS



DOM



Components everywhere~



Components everywhere~



Components everywhere~

Beranda IR Info Perseroan Info Finansial Tata Kelola Keterbukaan Info Saham Permintaan Info ID - EN

Hubungan Investor

Informasi kinerja Perseroan terkini dan transparan bagi investor dan masyarakat

www.bankbsi.co.id
Bank Syariah Indonesia Call 14040

PT Bank Syariah Indonesia Tbk berizin & diawasi oleh Otoritas Jasa Keuangan serta merupakan Peserta Penjaminan

Harga Saham

2.530 <small>▼ -20 (-0.78 %)</small>	Volume	11.556.900	Nama Emiten	Bank Syariah Indonesia
Tertunda 10 Menit Transaksi terakhir: 14 Mar 2025 16:14	Rentang Hari Ini	2.520 - 2.560	Kode Saham	BRIS

Komponent Pertama: Header



Beranda IR Info Perseroan Info Finansial Tata Kelola Keterbukaan Info Saham Permintaan Info ID - EN

Komponent Kedua: Slider

Hubungan Investor

Informasi kinerja Perseroan terkini dan transparan bagi investor dan masyarakat



www.bankbsi.co.id
Bank Syariah Indonesia Call 14040

PT Bank Syariah Indonesia Tbk berizin & diawasi oleh Otoritas Jasa Keuangan serta merupakan Peserta Penjaminan

Komponen Ketiga: Saham

Harga Saham

2.530 ¥ -20 (-0.78 %)

Tertunda 10 Menit
Transaksi terakhir: 14 Mar 2025 16:14

Volume	11.556.900	Nama Emiten	Bank Syariah Indonesia
Rentang Hari Ini	2.520 - 2.560	Kode Saham	BRIS

Pertanyaan:

**Coba berikan daftar komponen pada contoh website
yang diberikan**

Sekarang Saatnya Kita Hands-on!

2. Props, Children & Conditional Rendering

Elemen tambahan pada pembuatan komponen



Bahas Analogi: Props

Props = Paket Pesanan dalam Pengiriman



Restoran = Komponen utama

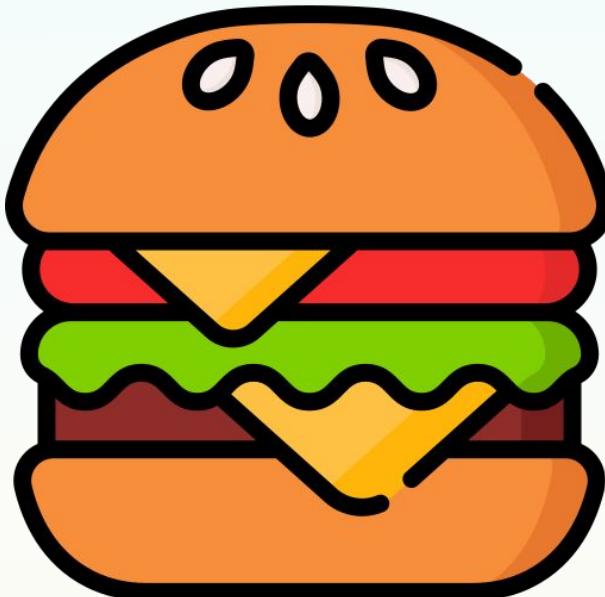
Driver pengantar = Props

Makanan yang dipesan = Data yang dikirim
lewat props

Contoh Penggunaan Props

```
function Welcome(props) {  
  return <h1>Hello, {props.name}!</h1>;  
}  
  
// Digunakan seperti ini:  
<Welcome name="Alice" /> // Output: Hello, Alice!
```

Bahas Analogi: Children



Roti = Komponen Utama

Isi Burger = Children:

- Jika kamu mau burger keju, kamu bisa isi dengan keju 
- Jika kamu mau burger ayam, kamu bisa isi dengan ayam 
- Jika kamu mau burger spesial, kamu bisa isi dengan keju, ayam, dan saus ekstra! 

Contoh Penggunaan Children

```
function Burger(props) {  
  return (  
    <div className="burger">  
      🍔 {props.children} 🍔  
    </div>  
  );  
}  
  
// Digunakan seperti ini:  
<Burger>Keju</Burger>    // Output: 🍔 Keju 🍔  
<Burger>Ayam</Burger>    // Output: 🍔 Ayam 🍔  
<Burger>Keju & Ayam</Burger> // Output: 🍔 Keju & Ayam 🍔
```

Terus kapan saatnya menggunakan Props dan Children?

Pakai Props Saat:

Gunakan **props** ketika kita ingin **memberikan informasi spesifik ke dalam komponen**.



Bayangkan kamu memesan **pizza** melalui aplikasi.

- Kamu harus memilih ukuran (**besar/sedang/kecil**)
- Kamu harus memilih topping (**pepperoni, keju, jamur, dll.**)

Ketika kamu memilih topping dan ukuran, **kamu tidak bisa mengubahnya setelah pizza selesai dibuat.**

Pakai **Children** Saat:

Gunakan **children** ketika kita ingin memasukkan **konten fleksibel** di dalam komponen.



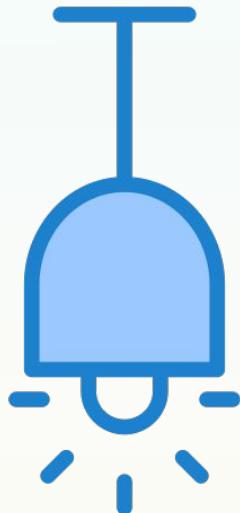
Bayangkan kamu punya **kotak hadiah** yang bisa diisi dengan apa saja.

- Kotaknya tetap sama (**komponen utama**)
- Isinya bisa berbeda (**children**), misalnya boneka, buku, atau coklat.

Dengan **children**, kita bisa membuat satu komponen yang isinya bisa berubah-ubah tanpa mengubah strukturnya.

Conditional Rendering

Bayangkan kamu punya **lampa ajaib** di kamar yang bisa menyala atau mati sendiri berdasarkan suatu kondisi!



- **Jika ada orang masuk ke kamar** → Lampu otomatis menyala 
- **Jika kamar kosong** → Lampu otomatis mati 

Di React, **Conditional Rendering** bekerja seperti lampu ajaib ini. Kita bisa menampilkan atau menyembunyikan sesuatu berdasarkan kondisi tertentu.

Contoh Conditional Rendering

```
function Lampu(props) {  
  if (props.adaOrang) {  
    return <h2>💡 Lampu Menyala!</h2>;  
  } else {  
    return <h2>🔴 Lampu Mati!</h2>;  
  }  
}  
  
// Cara menggunakannya:  
<Lampu adaOrang={true} /> // Output: 💡 Lampu Menyala!  
<Lampu adaOrang={false} /> // Output: 🔴 Lampu Mati!
```

3. State & Render

Elemen tambahan pada pembuatan komponen



State



State Analogy: Super Mario



State Analogy: Super Mario [TLDR]



Breakdown!!

Di pojok layar ada **skor** yang menunjukkan berapa banyak koin yang sudah kamu kumpulkan.

- Awalnya skor = 0
- Setiap kali Mario mengambil koin, skor bertambah!
- Setiap kali Mario kena musuh, skor bisa berkurang!

Nah, skor ini adalah **STATE**!

📌 State adalah data yang bisa berubah dan disimpan dalam sebuah komponen.

State Analogy: Score Bola

UEFA Champions League · 20 Feb

Full-time



3

-

1



Real Madrid

Man City

Aggregate: 6 - 3

Kylian Mbappé 4', 33', 61'



Nico González 90+2'

Render Analogy: Super Mario



Render Analogy: Super Mario [TLDR]



Breakdown!!

Sekarang bayangkan jika skor di layar tidak berubah saat Mario mengambil koin. Itu akan membingungkan, kan? 😕

Di React, saat state berubah, tampilan akan diperbarui secara otomatis!

Contoh Penggunaan State & Render

```
function Game() {  
  const [score, setScore] = React.useState(0); // State untuk skor  
  
  return (  
    <div>  
      <h2>🎮 Skor: {score}</h2>  
      <button onClick={() => setScore(score + 1)}>Ambil Koin! 🎁</button>  
    </div>  
  );  
}
```

4. Event & Navigation

Elemen tambahan pada pembuatan komponen



Event Analogy: Super Mario Kart



Event Analogy: Super Mario Kart [TLDR]



Breakdown!!

Bayangkan kamu sedang bermain **game balapan** 🏎️.

- Jika kamu **menekan tombol gas**, mobil akan melaju! 🚗💨
- Jika kamu **menekan tombol rem**, mobil akan berhenti! ⚡🔴
- Jika kamu **menekan tombol bel**, mobil akan berbunyi! 🔊

Ini sama seperti Event di React!

Ketika kita **mengklik tombol atau melakukan sesuatu**, React bisa menjalankan aksi tertentu.

Contoh Event Dalam React

```
function Mobil() {  
  function jalan() {  
    alert("Mobil Melaju! 🚗");  
  }  
  
  return <button onClick={jalan}>Tekan Gas! 🚗</button>;  
}
```

Navigation Analogy: Super Mario Bus



Event Analogy: Super Mario Bus [TLDR]



Breakdown!!

Sekarang bayangkan kamu sedang naik **bus** keliling **kota!** 🚍

- Jika ingin pergi ke **Taman Bermain**, kamu pilih rute ke **Taman Bermain** 🎡
- Jika ingin pergi ke **Kebun Binatang**, kamu pilih rute ke **Kebun Binatang** 🐆
- Jika ingin pulang, kamu pilih rute ke **Rumah** 🏠

Ini sama seperti Navigasi di React menggunakan React Router!

Kita bisa pindah dari **halaman satu ke halaman lain** dengan menekan tombol, seperti memilih rute di peta.

Contoh Navigation Dalam React

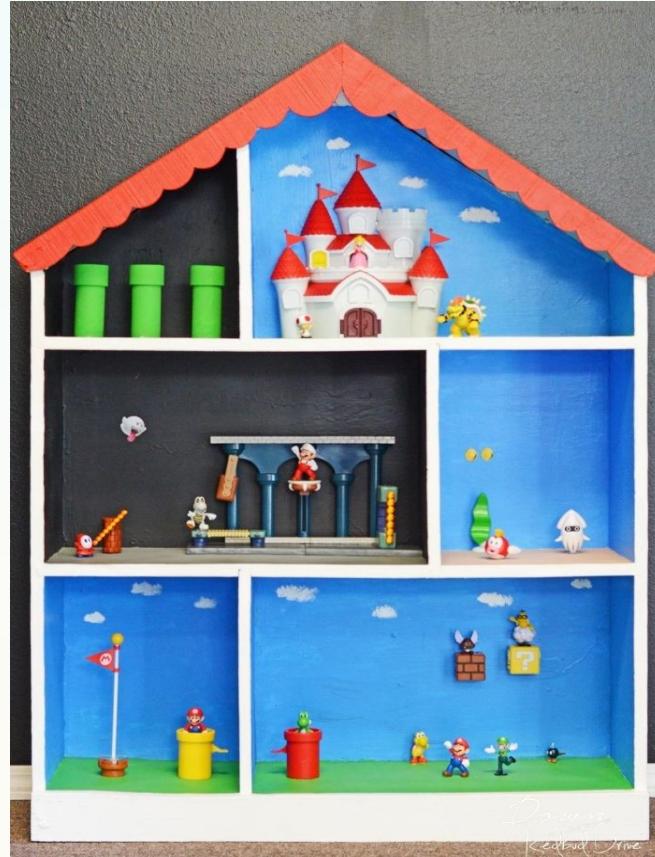
```
import { BrowserRouter as Router, Route, Link, Routes } from "react-router-dom";

function App() {
  return (
    <Router>
      <nav>
        <Link to="/taman">Pergi ke Taman 🌳</Link> |
        <Link to="/kebun">Pergi ke Kebun Binatang 🐻</Link>
      </nav>

      <Routes>
        <Route path="/taman" element={<h2>🌳 Selamat datang di Taman Bermain!</h2>} />
        <Route path="/kebun" element={<h2>🐻 Selamat datang di Kebun Binatang!</h2>} />
      </Routes>
    </Router>
  );
}


```

Layout Analogy: Bus Kota



Layout Analogy: Super Mario House [TLDR]



Bayangkan kamu punya **rumah besar** dengan banyak kamar! 

- Ruang tamu ada **sofa & TV** 
- Kamar tidur ada **kasur & lemari** 
- Dapur ada **kompor & kulkas** 

Di React, layout digunakan untuk menyusun tampilan halaman supaya lebih rapi!

Contoh Layouting Dalam React

```
function Layout(props) {
  return (
    <div>
      <header>Ini Header</header>
      <main>{props.children}</main>
      <footer>Ini Footer</footer>
    </div>
  );
}

function Home() {
  return <h2>Selamat Datang di Rumah!</h2>;
}

function App() {
  return (
    <Layout>
      <Home />
    </Layout>
  );
}
```

Thank You

React.js

Introduction

