KOMDIGI
Kementerian Komunikasi dan Digital
Republik Indonesia

digitalent

VSGA Vocational School Graduate Academy

PERTEMUAN 10-11

PELATIHAN JUNIOR WEB DEVELOPER

Menulis Kode Dengan Prinsip Sesuai Guidelines Dan Best Practice

(Nama Lokasi/Saluran Daring, Tanggal Penyampaian)

Course Definition

Deskripsi Singkat mengenai Topik
Menerapkan coding- guidelines dan best practices dalam penulisan program.

Tujuan Pelatihan
Peserta memiliki kemampuan dan keterampilan dalam menerapkan prinsip kode dengan baik sesuai aturan kesepakatan agar mudah dibaca, dikelola serta optimal dalam penggunaan memory.

Materi Yang akan disampaikan:
1. Menerapkan coding- guidelines dan best practices dalam penulisan program
2. Mendeksi Error Code dan cara mengatasinya
3. Menggunakan ukuran performansi dalam menuliskan kode sumber

Tugas :
1. **Mendeteksi error coding HTML dan cara mengatasinya**
2. **Mendeteksi error coding PHP dan cara mengatasinya**

Outcome/Capaian Pelatihan
Peserta mampu menerapkan coding-guidelines dan best practices dalam penulisan program

Deskripsi Unit Kompetensi terkait



MENTERI KETENAGAKERJAAN
REPUBLIK INDONESIA

KEPUTUSAN MENTERI KETENAGAKERJAAN
REPUBLIK INDONESIA
NOMOR 282 TAHUN 2016
TENTANG
PENETAPAN STANDAR KOMPETENSI KERJA NASIONAL INDONESIA
KATEGORI INFORMASI DAN KOMUNIKASI GOLONGAN POKOK AKTIVITAS PEMROGRAMAN, KONSULTASI KOMPUTER DAN KEGIATAN YANG BERHUBUNGAN DENGAN ITU (YBDI) BIDANG *SOFTWARE DEVELOPMENT*
SUBBIDANG PEMROGRAMAN

KODE UNIT : **J.620100.016.01**

JUDUL UNIT : **Menulis Kode dengan Prinsip sesuai *Guidelines* dan *Best Practices***

DESKRIPSI UNIT : Unit ini menentukan kompetensi, pengetahuan dan Sikap kerja yang diperlukan dalam menerapkan prinsip penulisan kode yang baik agar kode tersebut dapat dirawat (*maintainability*).

ELEMEN KOMPETENSI	KRITERIA UNJUK KERJA
1. Menerapkan *coding-guidelines* dan *best practices* dalam penulisan program (kode sumber)	1.1 Kode sumber dituliskan mengikuti ***coding-guidelines*** dan *best practices*. 1.2 Struktur program yang sesuai dengan konsep paradigmanya dibuat. 1.3 Galat/*error* ditangani.
2. Menggunakan ukuran performansi dalam menuliskan kode sumber	2.1 Efisiensi penggunaan *resources* oleh kode dihitung. 2.2 Kemudahan interaksi selalu di-implementasikan sesuai standar yang berlaku.

Learning Objectives

In this course you will:

1. Peserta mampu menghasilkan rekaman penerapkan coding- guidelines dan best practices dalam penulisan program
2. Peserta mampu menghasilkan rekaman penggunaan ukuran performansi dalam menuliskan kode sumber

Pre-Test

Berikan test tulis dalam multiple choice untuk mengukur pengetahuan peserta tentang:

1. Coding guidelines meliputi penamaan, penggunaan komentar, indentasi dalam pemrograman
2. Resources meliputi penggunaan memori dan lama eksekusi.
3. Efisiensi dalam kode sumber terkait dengan efisiensi Langkah proses (kecepatan) dan efisiensi penggunaan memori.
4. Paradigma Bahasa pemrograman seperti terstruktur atau beriorientasi objek.

1. Coding- guidelines dan best practices dalam penulisan program

Coding Guidelines: Ini adalah aturan dan standar yang diikuti para pengembang saat menulis kode. Tujuannya adalah untuk menciptakan konsistensi dalam kode yang ditulis oleh berbagai pengembang. Guidelines ini bisa mencakup:

1. Penamaan variabel dan fungsi yang konsisten dan deskriptif.
2. Penggunaan komentar untuk menjelaskan bagian kode yang kompleks.
3. Format penulisan kode, seperti indentasi dan penempatan kurung.
4. Penggunaan struktur kontrol seperti loop dan kondisional.
5. Mematuhi standar pemrograman untuk bahasa yang digunakan, seperti PEP 8 untuk Python.

Best Practices: Best practices adalah teknik atau metode yang dianggap ideal berdasarkan pengalaman dan riset dalam pengembangan perangkat lunak. Best practices membantu meningkatkan kualitas kode dan efisiensi proses pengembangan. Contoh best practices meliputi:

1. Menulis kode yang modular dan reusable.
2. Melakukan testing secara teratur dan menyeluruh.
3. Menggunakan version control systems seperti Git.
4. Menulis dokumentasi yang jelas untuk kode.
5. Menerapkan prinsip-prinsip desain perangkat lunak seperti SOLID atau DRY (Don't Repeat Yourself).
6. Menggunakan alat dan kerangka kerja yang tepat untuk meningkatkan produktivitas.
7. Mengikuti prinsip keamanan dalam pengkodean untuk menghindari kerentanan.

1.A Coding-Standard HTML

❖Struktur Dasar Dokumen HTML

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    <h1>This is a Heading</h1>
    <p>This is a paragraph.</p>
  </body>
</html>
```

Gunakan Lower Case untuk Nama

- Elemen dan atribut harus ditulis menggunakan huruf lower case

```
<!-- Correct -->
<input name="name" type="text" />

<!-- Wrong -->
<input name="name" TYPE="text" />
```

Tag Penutup

- Baik elemen yang kosong maupun yang tidak kosong, harus memiliki tag penutup yang sesuai.
- Elemen yang tidak kosong

```
<h1>My title</h1>
<p>Some text</p>
```

- Elemen yang kosong

```
<span></span>
```

Tag Penutup (2)

- Khusus untuk elemen dengan *single tag* harus menggunakan > di akhir statement.

```
<br>
<hr>
<img src="john.jpg" alt="John Doe" width="200" height="100">
```

Elemen Bersarang

Elemen bersarang (nested element) harus ditulis dengan benar, sesuai dengan susunan tag-nya.

Penulisan yang benar

```
<!-- Correct -->
<div>
  <p>Some text</p>
</div>
```

Penulisan yang salah

```
<!-- Wrong -->
<div>
  <p>Some text</div>
</p>
```

Nilai dari Atribut

- Menuliskan nilai (value) dari sebuah atribut harus menggunakan tanda petik dua [“ ”], walaupun nilai tersebut berupa nilai numerik.

```
<!-- Correct -->
<input name="age" type="text" size="3" />

<!-- Wrong -->
<input name=age type=text size=3 />
```

Indentasi

- Gunakan 2 spasi untuk indentasi kode.
- Gunakan indentasi secara konsisten untuk memudahkan dalam membaca kode.
- Ketika ada sebuah elemen yang didalamnya terdapat lebih dari satu baris kode, berikan indentasi pada konten elemen, antara tag awal dan tag akhir, untuk memudahkan dalam melihat di mana elemen dimulai dan berakhir.

```
<div class="container">
  <header class="header">
    <h1>Site Name<span></span></h1>
  </header>
  <!-- / header -->
  <hr>
  <nav class="navigation">
    <ul>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
    </ul>
  </nav>
  <!-- / navigation -->
</div>
<!-- / container -->
```

Encoding dan Charset

- Gunakan encoding standard untuk dokumen HTML anda, yaitu UTF-8 Normalization Form C (NFC)

```
<meta charset="utf-8" />
```

- Beberapa contoh untuk encoding character yang biasa digunakan

```
&amp;
&copy;
&raquo;
&gt;
```

HTML Anchors

❖Saat Anda ingin memberi tautan ke bagian di dalam dokumen HTML, gunakan atribut ID

```
<a href="#section">link</a>
<div id="section"></div>
```

❖Jika tidak mungkin menggunakan ID (misalnya karena platform ASP.NET), gunakan atribut *named anchors*

```
<a href="#section">link</a>
<a name="section"></a>
```

Komentar

❖Tuliskan komentar pada bagian akhir setelah tag penutup dari bagian HTML, hindari penulisan komentar pada awal bagian HTML.

```
<ol class="accessibility-nav">
  <li><a href="#navigation">Skip to navigation</a></li>
  <li><a href="#content">Skip to content</a></li>
  <li><a href="#sidebar">Skip to sidebar</a></li>
</ol>
<!-- / accessibility-nav -->

<p>
  <a href="#" title="Go to homepage"><em>Home</em></a>
</p>
<!-- / breadcrumb -->
```

Aksesibelitas

- Gunakan h1 - h6 untuk mengidentifikasi heading
- Gunakan elemen list (ul, ol, li) untuk mengelompokkan tautan
- Berikan definisi singkatan dengan menggunakan elemen abbr dan akronim
- Gunakan atribut bahasa pada elemen html untuk mengidentifikasi bahasa default suatu dokumen HTML

Table

- Gunakan elemen table <table></table> untuk menyajikan informasi dalam bentuk tabel
- Gunakan atribut scope untuk mengaitkan sel header dan sel data dalam tabel data
- Gunakan atribut summary dari elemen tabel untuk memberikan gambaran umum tentang data table

```
</tr><tr>
  <td>2.</td>
  <td scope="row">Clive Lloyd</td>
  <td>410-306-1420</td>
  <td>410-306-5400</td>
  <td>Baltimore</td>
</tr><tr>
  <td>3.</td>
  <td scope="row">Gordon Greenidge</td>
  <td>281-564-6720</td>
  <td>281-511-6600</td>
  <td>Houston</td>
```

Form

- Dalam sebuah form selalu sediakan tombol submit.
- Gunakan atribut alt pada gambar yang digunakan sebagai tombol kirim.
- Gunakan elemen label untuk mengaitkan label teks dengan kontrol formulir.
- Gunakan atribut judul untuk mengidentifikasi kontrol formulir ketika elemen label tidak dapat digunakan.

1.B Coding Standard dalam PHP

- PHP Coding Standard adalah kumpulan aturan dan pedoman untuk menulis kode PHP yang bersih, konsisten, dan mudah dibaca serta dipelihara oleh pengembang lain. Hal ini mencakup konvensi penamaan, gaya penulisan kode, penggunaan tanda kurung, indentasi, dan praktik terbaik lainnya yang memudahkan kerja sama dalam tim dan meningkatkan kualitas kode secara keseluruhan.
- PSR (PHP Standards Recommendation) adalah serangkaian rekomendasi standar yang dikembangkan oleh komunitas PHP Framework Interoperability Group (FIG). Standar-standar ini bertujuan untuk mempromosikan keseragaman dan interoperabilitas antara framework, pustaka, dan komponen yang ditulis dalam bahasa pemrograman PHP.

Stuktur Program PHP yang Paling Dasar

1. Menulis Kode HTML dan PHP

```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <title><?php echo "Belajar PHP" ?></title>
5.      </head>
6.      <body>
7.          <?php
8.              echo "saya sedang belajar PHP<br>";
9.              echo "<p>Belajar PHP hingga jadi master</p>";
10.         ?>
11.     </body>
12. </html>
```

Penjelasan

Kode PHP yang Anda berikan adalah sebuah dokumen HTML sederhana dengan skrip PHP yang di-embed untuk mengubah judul dan konten di dalam badan (body) dokumen. Berikut adalah penjelasan detilnya:

- **<!DOCTYPE html>**: Deklarasi ini mendefinisikan dokumen tersebut sebagai HTML5.
- **<html>**: Elemen root dari sebuah halaman HTML.
- **<head>**: Elemen head berisi informasi meta tentang dokumen.
  - **<title><?php echo "Belajar PHP" ?></title>**: Ini adalah judul dari dokumen HTML, yang diatur menggunakan PHP. Kode PHP di dalam tag **<title>** akan menghasilkan output "Belajar PHP" sebagai judul yang muncul di tab atau bar judul browser web.
- **<body>**: Elemen body berisi konten-konten dari dokumen HTML.
  - **<?php echo "saya sedang belajar PHP
"; ?>**: Skrip PHP pertama di dalam body mengeluarkan string "saya sedang belajar PHP" diikuti dengan tag **
** yang berarti baris baru.
  - **<?php echo "<p>Belajar PHP hingga jadi master</p>"; ?>**: Skrip PHP kedua mengeluarkan string "Belajar PHP hingga jadi master" yang dibungkus dengan tag paragraf (**<p>**).

2. Penulisan Statement dalam PHP

```
1. <?php
2. echo "ini statement 1";
3. echo "ini statement 2";
4. $a = $b + $c
```

3. Cara penulisan Case

PHP adalah bahasa pemrograman yang bersifat case sensitive. Artinya, huruf besar (kapital) dan huruf kecil akan dibedakan.

```
1. <?php
2. $nama = "Ali";
3. $NAMA = " Wardana";
4. ?>
```

Variabel $nama dan $NAMA adalah dua variabel yang berbeda. Mereka tidak sama.
Penulisan huruf besar dan kecil dalam program harus diperhatikan, karena bisa menyebabkan error bila salah.

4. Penulisan Komentar di PHP

Komentar di PHP dapat ditulis dengan dua cara:
Menggunakan tanda // untuk komentar satu baris;
Menggunakan tanda /* untuk komentar lebih dari satu baris.
Contoh:

```php
<?php

// ini adalah komentar
echo "Hello world";

/*
ini adalah komentar
yang lebih dari satu
baris
*/

?>
```

5. Penulisan Blok Program

Blok program merupakan kumpulan dari statement dan ekspresi.
Blok program di PHP dibungkus dengan kurung kurawal { ... }.

Contoh:

```php
if ($umur > 18){
    echo "Kamu sudah dewasa";
    echo "Selamat datang";
    echo "Kamu boleh minum kopi";
}
```

Program di atas adalah contoh blok kode *if* yang berisi tiga statement.

1.C. JQuery coding guidelines

1. Cara install / menggunakan JQuery
2. Definisi Variabel
3. Operator
4. Function
5. Selector
6. Event
7. Contoh Kasus membuat form dan hitung order

1. Definisi dan Install Jquery

jQuery adalah sebuah library JavaScript yang cepat, ringan, dan mudah digunakan. Tujuan utama jQuery adalah menyederhanakan penulisan kode JavaScript yang umumnya dibutuhkan dalam pengembangan website, seperti manipulasi DOM, event handling, animasi, AJAX, dan masih banyak lagi.

1. **Unduh jQuery:** Kunjungi situs resmi jQuery di https://jquery.com/ dan unduh versi terbaru jQuery. Anda dapat memilih untuk mengunduh versi produksi (production) atau versi pengembangan (development). Versi produksi direkomendasikan untuk digunakan di lingkungan produksi karena ukurannya lebih kecil dan sudah dioptimalkan. Sementara itu, versi pengembangan lebih cocok digunakan saat pengembangan atau debugging karena menyertakan komentar dan informasi tambahan. Tambahkan jQuery ke Proyek Anda:

1. **Tambahkan jQuery ke Proyek Anda:**

- Jika Anda menggunakan CDN (Content Delivery Network) jQuery, Anda cukup menambahkan tag <script> ke halaman HTML Anda, seperti berikut:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
```

- Jika Anda telah mengunduh file jQuery ke dalam proyek Anda, Anda bisa menyimpannya dalam direktori proyek dan menambahkannya ke dalam halaman HTML Anda, misalnya:

-
```
<script src="path/to/jquery.min.js"></script>
```

1. Definisi dan Install Jquery

3. Gunakan jQuery: Setelah Anda menambahkan jQuery ke halaman HTML Anda, Anda dapat mulai menggunakan fitur-fiturnya dengan menulis kode JavaScript yang memanfaatkannya. Misalnya:

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Contoh Penggunaan jQuery</title>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
</head>

<body>

    <button id="clickMe">Klik saya!</button>

    <script>
        $(document).ready(function () {
            $("#clickMe").click(function () {
                alert("Anda mengklik tombol!");
            });
        });
    </script>

</body>

</html>
```

Kode di samping menggunakan jQuery untuk menangani event klik pada sebuah tombol dan menampilkan pesan peringatan saat tombol tersebut diklik.

Dengan mengikuti langkah-langkah di atas, Anda dapat menginstal dan menggunakan jQuery dalam proyek web Anda.

1. Definisi variable

Deklarasi variabel dalam jQuery tidak berbeda dengan deklarasi variabel dalam JavaScript biasa. Anda dapat menggunakan tiga kata kunci var, let, atau const untuk mendeklarasikan variabel. Berikut adalah cara mendeklarasikan variabel dan contoh penggunaannya menggunakan jQuery:

```
// Mendeklarasikan variabel dengan var
var nama = "John";

// Penggunaan variabel dengan jQuery
$("#elementId").text(nama);
```

```
// Mendeklarasikan variabel dengan let
let usia = 25;

// Penggunaan variabel dengan jQuery
$("#elementId").text("Usia: " + usia);
```

```
// Mendeklarasikan variabel dengan const
const PI = 3.14;

// Penggunaan variabel dengan jQuery
$("#elementId").text("Nilai PI: " + PI);
```

2. Definisi variable

Berikut adalah perbedaan utama **antara var, let,** dan **const** dalam JavaScript secara umum, yang juga berlaku dalam konteks penggunaan jQuery:

var:
Variabel yang dideklarasikan dengan var memiliki lingkup (scope) yang berbeda tergantung pada di mana mereka dideklarasikan.
Variabel var bersifat hoisted, yang berarti deklarasinya akan didorong ke atas ruang lingkup (scope) dan dapat diakses sebelum variabel tersebut sebenarnya dideklarasikan.
Variabel var dapat dideklarasikan ulang dan diinisialisasi ulang.

let:
Variabel yang dideklarasikan dengan let memiliki lingkup blok (block scope), yang berarti variabel tersebut hanya dapat diakses di dalam blok di mana ia dideklarasikan.
Variabel let juga bersifat hoisted, tetapi hanya akan diberikan nilai undefined sampai titik di mana ia dideklarasikan dalam kode.
Variabel let dapat diinisialisasi ulang, tetapi tidak dapat dideklarasikan ulang dalam lingkup yang sama.

const:
Variabel yang dideklarasikan dengan const juga memiliki lingkup blok (block scope) seperti let.
Namun, variabel const harus diinisialisasi dengan nilai saat dideklarasikan dan nilainya tidak dapat diubah setelahnya. Ini berlaku baik untuk primitif maupun objek.
Variabel const tidak dapat dideklarasikan ulang atau diinisialisasi ulang.
Dalam praktiknya, baik Anda menggunakan var, let, atau const dalam pengembangan dengan jQuery tergantung pada kebutuhan spesifik proyek Anda dan preferensi pengodean Anda. Namun, sebagai praktik yang baik, disarankan untuk menggunakan const untuk variabel yang nilainya tidak akan berubah, let untuk variabel yang nilainya mungkin berubah, dan var hanya digunakan dalam kasus tertentu atau jika Anda bekerja di lingkungan di mana let dan const tidak didukung.

3. Operator

Dalam jQuery, Anda dapat menggunakan operator seperti yang biasa digunakan dalam JavaScript. Operator-operator ini digunakan untuk operasi matematika, perbandingan, logika, dan lain-lain. Berikut adalah beberapa contoh operator yang sering digunakan dalam jQuery:

Operator Penugasan:
Pengisian (=): a = b
Penambahan (+=): a += b (sama dengan a = a + b)
Pengurangan (-=): a -= b (sama dengan a = a - b)
Perkalian (*=): a *= b (sama dengan a = a * b)
Pembagian (/=): a /= b (sama dengan a = a / b)
Modulus (%=): a %= b (sama dengan a = a % b)

```
var total = 10;
total += 5; // total sekarang akan menjadi 15
```

Operator Matematika:
Penambahan (+): a + b
Pengurangan (-): a - b
Perkalian (*): a * b
Pembagian (/): a / b
Modulus (%): a % b

```
var total = 10 + 5; // total akan menjadi 15
```

Operator Logika:
AND (&&): a && b
OR (||): a || b
NOT (!): !a

```
var a = true;
var b = false;
var hasil = a && b; // hasil akan menjadi false
```

Operator Perbandingan:
Sama dengan (==): a == b
Tidak sama dengan (!=): a != b
Lebih besar dari (>): a > b
Lebih kecil dari (<): a < b
Lebih besar dari atau sama dengan (>=): a >= b
Lebih kecil dari atau sama dengan (<=): a <= b

```
var x = 5;
var y = 10;
var hasil = x > y; // hasil akan menjadi false
```

Operator Ternary:
Ternary (?:): condition ? expr1 : expr2

```
var angka = 10;
var hasil = (angka > 5) ? "Lebih besar dari lima" : "Tidak lebih besar dari lima";
```

4. Function

Untuk membuat dan menggunakan fungsi dalam jQuery, Anda dapat menggunakan sintaksis dasar JavaScript untuk mendefinisikan fungsi, dan kemudian menggunakan fungsi tersebut dalam konteks pengembangan jQuery. Berikut adalah cara membuat dan menggunakan fungsi dalam jQuery:

1. Membuat/Mendefinisikan Fungsi di jQuery:
Anda dapat membuat fungsi dalam jQuery dengan menggunakan sintaksis dasar JavaScript. Berikut adalah contoh cara mendefinisikan fungsi:

```
// Mendefinisikan fungsi dengan jQuery
function myFunction() {
    // Isi fungsi di sini
    alert("Hello, this is my function!");
}
```

Anda juga dapat menggunakan cara lain untuk mendefinisikan fungsi dalam jQuery, seperti menggunakan ekspresi fungsi atau menggunakan fungsi anonim:

```
// Mendefinisikan fungsi dengan ekspresi fungsi
var myFunction = function() {
    // Isi fungsi di sini
    alert("Hello, this is my function!");
};

// Mendefinisikan fungsi anonim
$(document).ready(function() {
    // Isi fungsi di sini
    alert("Hello, this is my function!");
});
```

Menggunakan Fungsi di jQuery:
Setelah Anda mendefinisikan fungsi, Anda dapat memanggilnya di berbagai bagian kode jQuery atau JavaScript. Berikut adalah contoh cara menggunakan fungsi dalam jQuery:

```
// Memanggil fungsi yang telah didefinisikan
myFunction();
```

5. Selector

Selektor (selector) dalam jQuery digunakan untuk memilih elemen-elemen HTML pada halaman web agar dapat dimanipulasi dengan JavaScript. Ada banyak jenis selektor yang dapat Anda gunakan dalam jQuery, yang memungkinkan Anda untuk memilih elemen-elemen berdasarkan ID, kelas, tipe, atribut, dan hubungan hierarki mereka dalam DOM (Document Object Model). Berikut adalah beberapa selektor yang sering digunakan dalam pengembangan web:

1. **Selektor Berdasarkan ID**:
   - Memilih elemen berdasarkan ID unik mereka.
   - Format: `` `$("#idElement")` ``
   - Contoh: `` `$("#myElement")` ``

2. **Selektor Berdasarkan Kelas**:
   - Memilih elemen-elemen berdasarkan kelas CSS mereka.
   - Format: `` `$(".className")` ``
   - Contoh: `` `$(".myClass")` ``

3. **Selektor Berdasarkan Tipe**:
   - Memilih elemen-elemen berdasarkan jenis tag HTML mereka.
   - Format: `` `$("tagName")` ``
   - Contoh: `` `$("div")`, `$("p")`, `$("input")` ``

4. **Selektor Berdasarkan Atribut**:
   - Memilih elemen-elemen berdasarkan nilai atribut HTML mereka.
   - Format: `` `$("[attributeName='value']")` ``
   - Contoh: `` `$("[name='username']")`, `$("[data-type='button']")` ``

6. event

Event di jQuery adalah tindakan atau peristiwa yang terjadi pada elemen HTML tertentu di halaman web, seperti klik mouse, mengetik pada keyboard, menggulir halaman, dan lain-lain. jQuery memungkinkan Anda menangani event ini dengan mudah menggunakan metode seperti .on(), .click(), .hover(), dan sebagainya.

Selector ID:
Misalkan kita memiliki elemen dengan ID #myButton yang merupakan sebuah tombol, dan kita ingin menambahkan event click ke tombol tersebut:

```
<button id="myButton">Click Me!</button>
```

Kemudian, kita dapat menambahkan event click ke tombol tersebut dengan jQuery:

```
$("#myButton").on("click", function() {
    alert("Tombol dengan ID 'myButton' diklik!");
});
```

Selector Class :
Misalkan kita memiliki beberapa tombol dengan class .myButton dan masing-masing tombol memiliki atribut data dengan nama action yang menentukan aksi yang akan diambil ketika tombol diklik:

```
<button class="myButton" data-action="save">Simpan</button>
<button class="myButton" data-action="edit">Edit</button>
<button class="myButton" data-action="delete">Hapus</button>
```

Kemudian, kita ingin menambahkan event onclick ke semua tombol dengan class .myButton yang memiliki atribut data action dengan value save menggunakan jQuery:

```
// Menambahkan event onclick menggunakan jQuery dengan selector Class dan filter den
$(".myButton[data-action='save']").on("click", function() {
    alert("Tombol dengan class 'myButton' dan data-action='save' diklik!");
});
```

6. event

Selector Attribut Name :

Misalkan kita memiliki beberapa tombol dengan attribute name action yang menentukan aksi yang akan diambil ketika tombol diklik:

```
<button name="action" value="save">Simpan</button>
<button name="action" value="edit">Edit</button>
<button name="action" value="delete">Hapus</button>
```

Kemudian, kita ingin menambahkan event onclick ke semua tombol dengan attribute name action menggunakan jQuery:

```
// Menambahkan event onclick menggunakan jQuery dengan selector attribute name
$("button[name='action']").on("click", function() {
    var action = $(this).attr("value");
    alert("Tombol dengan attribute name 'action' dan value '" + action + "' diklik!");
});
```

Dalam contoh di atas, kita menggunakan selector attribute name button[name='action'] untuk memilih semua tombol dengan attribute name action. Saat tombol tersebut diklik, fungsi yang diberikan akan dijalankan dan menampilkan pesan peringatan yang menyertakan value dari attribute value dari tombol yang diklik.

7. Contoh Kasus

Contoh Kasus Membuat Form Order Makanan

- Kita memiliki form pemesanan makanan sederhana dengan beberapa item makanan seperti Nasi Goreng, Ayam Goreng, Es Teh, dan Kopi. Setiap item memiliki harga dan input field untuk mengisi jumlah pesanan.
- Terdapat tombol "Hitung Total" yang ketika diklik, akan menghitung total harga dari semua pesanan yang diisi.
- jQuery digunakan untuk menangani event saat tombol "Hitung Total" diklik. Script akan mengambil harga dan jumlah pesanan untuk setiap item, menghitung total harga pesanan, dan menampilkan hasilnya.





7. Contoh Kasus (jawaban)

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Form Pemesanan Makanan</title>
</head>
<body>
    <h2>Form Pemesanan Makanan</h2>
    <table>
        <tr>
            <th>Nama Makanan</th>
            <th>Harga</th>
            <th>Jumlah</th>
        </tr>
        <tr class="item">
            <td>Nasi Goreng</td>
            <td class="harga">Rp10.000</td>
            <td><input type="number" class="jumlah" value="0" title="Jumlah Nasi Goreng"></td>
        </tr>
        <tr class="item">
            <td>Ayam Goreng</td>
            <td class="harga">Rp12.000</td>
            <td><input type="number" class="jumlah" value="0" title="Jumlah Ayam Goreng"></td>
        </tr>
        <tr class="item">
            <td>Es Teh</td>
            <td class="harga">Rp2.000</td>
            <td><input type="number" class="jumlah" value="0" title="Jumlah Es Teh"></td>
        </tr>
```

7. Contoh Kasus (jawaban)

```
        <tr class="item">
            <td>Kopi</td>
            <td class="harga">Rp3.000</td>
            <td><input type="number" class="jumlah" value="0" title="Jumlah Kopi"></td>
        </tr>
    </table>

    <!-- Tombol Hitung Total -->
    <button id="hitungTotal">Hitung Total</button>

    <!-- Hasil Total Pesanan -->
    <div id="totalPesanan"></div>

    <!-- Script JavaScript -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $(document).ready(function () {
            // Event handler untuk tombol "Hitung Total"
            $("#hitungTotal").click(function () {
                var total = 0;
                $(".item").each(function () {
                    var harga = parseInt($(this).find(".harga").text().replace("Rp", "").replace(".", ""));
                    var jumlah = parseInt($(this).find(".jumlah").val());
                    total += harga * jumlah;
                });
                $("#totalPesanan").text("Total Pesanan: " + total.toFixed(0));
            });
        });
    </script>
</body>
</html>
```

7. Contoh Kasus (Penjelasan)

1. $(document).ready(function() {...});: Ini adalah cara untuk menunggu hingga semua elemen HTML telah dimuat sebelum menjalankan kode JavaScript di dalamnya. Ini memastikan bahwa semua elemen yang kita tuju telah tersedia sebelum kita mulai berinteraksi dengan mereka.

1. $("#hitungTotal"): Ini adalah selector jQuery yang menargetkan elemen dengan ID hitungTotal. Ini adalah tombol "Hitung Total" yang akan memicu perhitungan saat diklik.

1. .click(function() {...});: Ini adalah metode jQuery yang menambahkan event handler untuk event "click" pada elemen yang dipilih. Fungsi yang diberikan akan dijalankan setiap kali elemen tersebut diklik.

1. .each(function() {...});: Ini adalah metode jQuery yang digunakan untuk mengulangi setiap elemen dalam koleksi yang dipilih. Di sini, kita menggunakan .each untuk mengulangi setiap item makanan dalam tabel.

1. $(this): Ini merujuk pada elemen yang sedang diulangi saat pengulangan. Dalam konteks ini, $(this) merujuk pada elemen .item yang sedang diproses.

1. .find(".harga"): Ini adalah metode jQuery yang mencari elemen dengan kelas .harga di dalam elemen yang dipilih ($(this)). Dalam konteks ini, kita mencari elemen harga di dalam setiap baris item makanan.

1. .text(): Ini adalah metode jQuery yang mengambil teks dari elemen yang dipilih. Di sini, kita mengambil teks dari elemen harga untuk mendapatkan nilai harga makanan.

7. Contoh Kasus (Penjelasan)

8. .replace("Rp", "").replace(".", ""): Ini adalah metode JavaScript untuk menghapus karakter "Rp" dan titik dari harga yang diambil dari elemen .harga. Ini dilakukan agar harga dapat diubah menjadi angka yang dapat dihitung.

8. parseInt(): Ini adalah fungsi JavaScript yang mengonversi string menjadi bilangan bulat. Kami menggunakannya di sini untuk mengonversi harga makanan dari string menjadi angka bulat.

8. parseInt($(this).find(".jumlah").val()): Ini adalah cara untuk mendapatkan nilai jumlah pesanan untuk setiap item makanan. Kami mencari elemen input dengan kelas .jumlah di dalam setiap baris item makanan dan mengambil nilai yang dimasukkan oleh pengguna.

8. total += harga * jumlah;: Di sini, kami menghitung total pesanan dengan menambahkan jumlah harga makanan (harga dikalikan dengan jumlah pesanan) ke total sebelumnya.

8. $("#totalPesanan").text("Total Pesanan: " + total.toFixed(0));: Akhirnya, kami menampilkan total pesanan di dalam elemen dengan ID totalPesanan. Kami menggunakan metode .text() untuk mengatur teks di dalam elemen tersebut menjadi string yang berisi total pesanan. Kami menggunakan .toFixed(0) untuk membulatkan total menjadi bilangan bulat tanpa desimal.

2. Cara menangani Error Code

Cakupan bahasan:

1. Error yang sering muncul di HTML dan cara mengatasinya
2. Error yang sering muncul di PHP dan cara mengatasinya

Error yang sering muncul di HTML dan cara mengatasinya

1. **Tag Tidak Tertutup**
   Contoh: `<p>Paragraf ini tidak pernah ditutup.`
   Cara Mengatasi: Pastikan setiap tag pembuka memiliki tag penutup yang sesuai.
   `<p>Paragraf ini sekarang ditutup.</p>`

1. **Atribut Tanpa Tanda Kutip**
   Contoh: `<a href=http://example.com>Link</a>`
   Cara Mengatasi: Bungkus nilai atribut dengan tanda kutip.
   `<a href="http://example.com">Link</a>`

1. **Elemen Bersarang yang Salah**
   Contoh: `<b><i>Text</b></i>`
   Cara Mengatasi: Pastikan elemen bersarang ditutup dalam urutan yang benar.
   `<b><i>Text</i></b>`

1. **Menggunakan Elemen yang Tidak Sesuai Standar atau Usang**
   Contoh: `<font size="3">Text</font>` (elemen `<font>` sudah usang)
   Cara Mengatasi: Gunakan CSS untuk styling.
   `<span style="font-size: 16px;">Text</span>`

1. **Kesalahan DOCTYPE**
   Contoh: Kesalahan atau ketiadaan deklarasi DOCTYPE.
   Cara Mengatasi: Pastikan dokumen HTML dimulai dengan DOCTYPE yang benar.
   `<!DOCTYPE html>`

2.1 Error yang sering muncul di PHP dan cara mengatasinya

1. Parse Error (Syntax Error)
Error ini terjadi ketika ada kesalahan sintaks dalam kode PHP. Biasanya karena kekurangan tanda seperti titik koma (;), kurung, tanda kutip, atau kesalahan pengetikan lainnya.

```
<?php
echo "Hello, world"
?>
```

Cara Mengatasi:
Periksa dan tambahkan titik koma (;) yang hilang di akhir pernyataan **echo**.

```
<?php
echo "Hello, world";
?>
```

2. Fatal Error

Fatal error terjadi ketika PHP tidak dapat melanjutkan proses eksekusi skrip karena adanya kesalahan seperti memanggil fungsi yang tidak didefinisikan atau mengakses properti dari non-objek.

```php
<?php
0 references
function greeting() {
    echo "Hello, world!";
}

greet(); // Fungsi greet() tidak didefinisikan
?>

```

Fatal error: Uncaught Error: Call to undefined function greet() in C:\laragon\www\jwd2024\contoh-error\error1.php on line *6*

Error: Call to undefined function greet() in C:\laragon\www\jwd2024\contoh-error\error1.php on line *6*

Call Stack

#	Time	Memory	Function	Location
1	0.4207	406904	{main}( )	...\error1.php:0

Cara mengatasi Fatal Error

Pastikan nama fungsi yang dipanggil sudah benar.

php Copy code

```php
<?php
function greeting() {
    echo "Hello, world!";
}

greeting(); // Memanggil fungsi greeting() yang sudah didefinisikan
?>
```

3. Warning

Warning biasanya terjadi saat PHP menemukan sesuatu yang bisa menjadi masalah, tapi tidak menghentikan eksekusi skrip. Contohnya, menginclude file yang tidak ada.

Contoh:

```php
<?php
include("file_tidak_ada.php");
?>
```

Cara Mengatasi: Periksa dan pastikan file yang ingin di-include memang ada dan lokasinya benar.

4. Notice

Notice adalah pesan kesalahan ringan yang menginformasikan kebiasaan pemrograman yang buruk atau potensi masalah, seperti **penggunaan variabel yang belum didefinisikan.**

Contoh:

```php
<?php
echo $variabelBelumDidefinisikan;
?>
```

Notice: Undefined variable: variabelBelumDidefinisikan in C:\laragon\www\jwd2024\contoh-error\warning.php on line 2

Call Stack

#	Time	Memory	Function	Location
1	0.4193	406176	{main}( )	...\warning.php:0

Cara Mengatasinya:
Definisikan variabel sebelum digunakan.

```php
<?php
$variabelBelumDidefinisikan = "Ini adalah contoh.";
echo $variabelBelumDidefinisikan;
?>
```

3. Penggunaan ukuran performansi dalam menuliskan kode sumber di PHP

Dalam pengembangan PHP, performansi adalah aspek penting yang perlu diperhatikan.
Berikut adalah beberapa ukuran performansi yang umum digunakan:

1. **Waktu Eksekusi**
   Mengukur berapa lama waktu yang diperlukan untuk menjalankan script PHP. Kode yang dioptimalkan akan memiliki waktu eksekusi yang lebih singkat.
   Contoh:

```
1.  $startTime = microtime(true);
2.  // Blok kode yang ingin diukur performansinya
3.  for ($i = 0; $i < 1000; $i++) {
    // Proses yang memakan waktu
    }
4.  $endTime = microtime(true);
5.  $executionTime = $endTime - $startTime;
6.  echo "Waktu eksekusi adalah: " . $executionTime . " detik";
```

Penjelasan code

1.$startTime = microtime(true);

Di sini kita menggunakan fungsi **microtime()** dengan parameter **true** untuk mendapatkan waktu saat ini dalam bentuk **float**, yang merupakan representasi waktu saat ini dalam detik dan mikrodetik. Ini menandai waktu mulai eksekusi blok kode.

2.// Blok kode yang ingin diukur performansinya

Ini adalah tempat Anda akan meletakkan blok kode yang performansinya ingin Anda ukur. Dalam contoh ini, ada loop **for** yang berjalan sebanyak 1000 kali.

3.for ($i = 0; $i < 1000; $i++) {
// Proses yang memakan waktu
}

Loop **for** ini akan melakukan iterasi dari 0 sampai 999. Di dalamnya, seharusnya terdapat kode yang ingin diukur, seperti pemrosesan data atau fungsi yang memakan waktu eksekusi.

4.$endTime = microtime(true);

Setelah blok kode dijalankan, kita mencatat waktu sekali lagi untuk menandai waktu akhir eksekusi.

5.$executionTime = $endTime - $startTime;

Kita menghitung waktu eksekusi dengan mengurangkan waktu mulai dari waktu akhir, yang memberikan durasi eksekusi blok kode dalam detik.

6.echo "Waktu eksekusi adalah: " . $executionTime . " detik";

Terakhir, kita mencetak waktu eksekusi ke layar. Ini akan menampilkan berapa lama waktu yang diperlukan untuk menjalankan blok kode tersebut dalam detik.

2. Penggunaan Memori

Mengukur jumlah memori yang digunakan oleh script PHP. Kode yang efisien akan menggunakan memori seminimal mungkin.

Contoh:

```
<?php

    echo "Penggunaan memori sebelum: " . memory_get_usage() . " bytes \n";

    // Blok kode yang ingin diukur konsumsi memorinya
    $array = range(1, 100000);

    echo "Penggunaan memori setelah: " . memory_get_usage() . " bytes \n";
?>
```

Penjelasan code

1.echo "Penggunaan memori sebelum: " . memory_get_usage() . " bytes \n";
Fungsi **memory_get_usage()** digunakan untuk mendapatkan jumlah memori yang saat ini dialokasikan untuk skrip PHP Anda, dinyatakan dalam byte. Ini dicetak ke layar sebagai penggunaan memori sebelum eksekusi blok kode tertentu. **\n** adalah karakter escape yang digunakan untuk membuat baris baru di output terminal atau command line interface.

1.// Blok kode yang ingin diukur konsumsi memorinya
Ini adalah komentar yang menunjukkan bahwa blok kode berikutnya adalah bagian yang ingin diukur konsumsi memorinya.

1.$array = range(1, 100000);
Fungsi **range()** digunakan untuk membuat sebuah array yang berisi nilai-nilai dalam rentang yang diberikan. Dalam hal ini, array akan diisi dengan angka-angka dari 1 sampai 100000. Ini akan menggunakan sejumlah memori dan Anda dapat melihat perbedaan dalam konsumsi memori sebelum dan setelah array ini dibuat.

1.echo "Penggunaan memori setelah: " . memory_get_usage() . " bytes \n";
Setelah array dibuat, **memory_get_usage()** dipanggil lagi untuk mendapatkan jumlah memori yang digunakan setelah array tersebut diinisialisasi. Ini juga dicetak ke layar sehingga Anda dapat melihat perbedaan dalam penggunaan memori sebelum dan sesudah array dibuat.

TUGAS 1

1. Tulislah atau copas code dibawah ini yang terdiri dari 3 file:
    a) Form-daftar.php (untuk tampilan front end isian pendaftaran)
    b) Proses-pendaftaran-2.php (penyimpanan data ke database, tabel
    c) Koneksi.php
    d) Script tabel calon_siswa.sql
2. Jalankan program tersebut, lalu deteksi error apa saja dan cara memperbaikinya
3. Uraikan error message yang teridentifikasi dan berikan penjelasan dan jawabannya

Referensi

1. https://www.w3schools.com/php/php_syntax.asp
2. https://www.w3schools.com/js/default.asp
3. https://www.petanikode.com/php-sintak/
4. Tutorial belajar PHP https://www.youtube.com/watch?v=TaBWhb5SPfc
5. Tutorial Java Script https://www.youtube.com/watch?v=mD6uSGSjgr4&list=PLc6SEcJkQ6DwiGCnVdCbWLqo6ceICD_4b
6. Tutorial Belajar Jquery https://www.youtube.com/watch?v=SXNfn1JCkGQ&list=PLce3Eyp7oY98VSAcerueAIOEsHfiszWxI

Tools

1. Teks Editor (sublime text / visual studio code / notepad ++)
2. XAMPP minimal versi 2.7
3. Web Browser (Google Chrome / safari / internet explorer)

…

…

Summary

1. Untuk menuliskan kode program perlu memperhatikan coding-guidelines dan best practices yang dapat dijadikan acuan
2. Coding guidelines meliputi penamaan, penggunaan komentar, indentasi yang berlaku di organisasi
3. Terdapat jenis-jenis error dalam membuat kode pemrograman yang harus mampu diperbaiki oleh junior web developer
4. Dalam menulis program perlu memperhatikan efisiensi penggunaan resources seperti utilisasi memory dan CPU Ketika program dijalankna

KOMDIGI
Kementerian Komunikasi dan Digital
Republik Indonesia

digitalent

Terima Kasih

