

HASIL TUGAS CLIENT SIDE

PEMROGRAMAN BERBASIS WEB

1. Materi

Dalam web development, client-side berarti segala sesuatu yang dilakukan atau dijalankan di sisi pengguna (komputer pengguna).

Client-side Scripting adalah bahasa pemrograman web yang pengolahan datanya dilakukan oleh komputer pengguna/pengunjung. Jadi, ketika seseorang berkunjung ke sebuah web, komputernya akan mendownload data/script yang bersifat client-side di web tersebut.

Client-side script meliputi kode-kode yang ditampilkan ketika anda mengklik kanan pada sebuah halaman web dan melihat sumber halaman (View Page Source). Contoh dari Client-side Script adalah HTML, CSS, JavaScript dan XML.

Dasar Teori

Cara menyambungkan Javascript ke HTML

Untuk menyambungkan JavaScript ke dalam dokumen HTML, Anda dapat menggunakan tag script di dalam elemen head atau body dokumen HTML. Berikut adalah contoh kode dasar:

Masukkan kode JavaScript di dalam tag script:

html

Copy code

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>Contoh Penyambungan Javascript ke HTML</title>
```

```
<script>
```

```
// Masukkan kode JavaScript di sini
```

```
</script>
```

```
</head>
```

```
<body>

  <!-- Isi dokumen HTML di sini -->

</body>

</html>
```

Sambungkan file JavaScript eksternal dengan menggunakan atribut src di dalam tag script:

html

Copy code

```
<!DOCTYPE html>

<html>

  <head>

    <title>Contoh Penyambungan Javascript ke HTML</title>

    <script src="script.js"></script>

  </head>

  <body>

    <!-- Isi dokumen HTML di sini -->

  </body>

</html>
```

Pastikan untuk menempatkan file JavaScript eksternal pada direktori yang benar dan menyesuaikan nama file dengan benar. Jika Anda ingin menggunakan framework JavaScript tertentu seperti React atau Vue, Anda dapat menggunakan cara yang lebih spesifik untuk menyambungkan JavaScript ke dalam dokumen HTML.

Deklarasi Variabel

a. Var

Deklarasi variabel menggunakan kata kunci var pada JavaScript digunakan untuk membuat variabel dan memberi tahu interpreter JavaScript bahwa kita akan menggunakan nama variabel tersebut. Variabel dapat berisi nilai yang berbeda selama program dijalankan.

Berikut adalah contoh penggunaan deklarasi variabel menggunakan var:

javascript

Copy code

```
var nama = "John";  
var umur = 25;  
var sudahMenikah = false;  
console.log(nama); // output: John  
console.log(umur); // output: 25  
console.log(sudahMenikah); // output: false
```

Dalam contoh di atas, kita membuat tiga variabel: nama, umur, dan sudahMenikah. nama adalah variabel bertipe data string, umur adalah variabel bertipe data number, dan sudahMenikah adalah variabel bertipe data boolean.

Variabel yang dideklarasikan dengan var dapat diakses dan dimodifikasi di mana saja dalam program JavaScript. Variabel yang dideklarasikan tanpa nilai akan memiliki nilai undefined secara default.

javascript

Copy code

```
var a;  
console.log(a); // output: undefined
```

Namun, perlu diingat bahwa variabel yang dideklarasikan dengan var memiliki scope fungsi. Artinya, jika variabel dideklarasikan di dalam fungsi, variabel

javascript

Copy code

```
var a;  
console.log(a); // output: undefined
```

Namun, perlu diingat bahwa variabel yang dideklarasikan dengan var memiliki scope fungsi. Artinya, jika variabel dideklarasikan di dalam fungsi, variabel tersebut hanya dapat diakses dari dalam fungsi tersebut.

```
javascript  
Copy code  
function tes() {  
  var b = 2;  
}
```

```
console.log(b); // output: error
```

Dalam contoh di atas, variabel b hanya dapat diakses dari dalam fungsi tes(). Jika kita mencoba mengakses variabel b dari luar fungsi, maka akan muncul error.

Let

let adalah kata kunci dalam JavaScript yang digunakan untuk mendeklarasikan variabel yang nilainya dapat diubah atau diinisialisasi ulang di masa depan. Variabel yang dideklarasikan dengan let memiliki scope block-level, yang berarti bahwa variabel hanya dapat diakses di dalam blok di mana variabel tersebut dideklarasikan.

Berikut adalah contoh penggunaan let untuk deklarasi variabel:

```
javascript  
Copy code  
let nama = "John";  
nama = "Doe";  
console.log(nama); // output: Doe
```

Dalam contoh di atas, kita membuat variabel nama menggunakan let. Setelah itu, kita mengubah nilainya menjadi "Doe". Kita dapat melakukannya karena variabel yang dideklarasikan menggunakan let dapat diubah nilainya.

Variabel yang dideklarasikan dengan `let` hanya dapat diakses di dalam blok di mana variabel tersebut dideklarasikan. Blok adalah bagian dari kode yang diapit oleh kurung kurawal `{ }`. Sebagai contoh:

javascript

Copy code

```
let x = 10;
if (x > 5) {
  let y = 5;
  console.log(x); // output: 10
  console.log(y); // output: 5
}
console.log(x); // output: 10
console.log(y); // output: error
```

Dalam contoh di atas, variabel `x` dideklarasikan menggunakan `const` di dalam blok `if`. Variabel `x` hanya dapat diakses di dalam blok `if` tersebut, sehingga jika kita mencoba mengakses variabel `x` di luar blok `if`, maka akan muncul error.

Function dan Kondisi

Fungsi di JavaScript

Fungsi adalah blok kode yang dapat digunakan kembali dalam program. Dalam JavaScript, fungsi dapat didefinisikan dengan menggunakan kata kunci `function`. Berikut adalah contoh fungsi sederhana yang menerima dua argumen dan mengembalikan hasil penjumlahannya:

javascript

Copy code

```
function tambah(a, b) {
  return a + b;
}
```

Dalam contoh di atas, kita mendefinisikan fungsi `tambah` dengan dua argumen `a` dan `b`. Fungsi ini mengembalikan hasil penjumlahan dari dua argumen tersebut.

Kita dapat memanggil fungsi tambah dengan memberikan dua argumen:

javascript

Copy code

```
let hasil = tambah(2, 3);  
console.log(hasil); // output: 5
```

Dalam contoh di atas, kita memanggil fungsi tambah dengan memberikan dua argumen 2 dan 3. Hasil penjumlahan dari dua argumen tersebut kemudian disimpan dalam variabel hasil, yang kemudian dicetak menggunakan console.log.

Kondisi di JavaScript

Kondisi adalah bagian dari program yang menentukan apakah blok kode tertentu akan dijalankan atau tidak, berdasarkan kondisi yang diberikan. Dalam JavaScript, kita dapat menggunakan beberapa kata kunci untuk membuat kondisi:

if

if adalah kata kunci yang digunakan untuk mengeksekusi blok kode jika kondisi tertentu terpenuhi. Berikut adalah contoh penggunaan if:

javascript

Copy code

```
let x = 10;  
if (x > 5) {  
  console.log("x lebih besar dari 5");  
}
```

Dalam contoh di atas, kita menggunakan if untuk mengecek apakah nilai x lebih besar dari 5. Jika kondisi terpenuhi, maka blok kode di dalam kurung kurawal akan dieksekusi.

else

else adalah kata kunci yang digunakan untuk mengeksekusi blok kode jika kondisi pada if tidak terpenuhi. Berikut adalah contoh penggunaan if-else:

javascript

Copy code

```
let x = 3;  
if (x > 5) {  
  console.log("x lebih besar dari 5");  
} else {  
  console.log("x kurang dari atau sama dengan 5");
```

Dalam contoh di atas, kita menggunakan if-else untuk mengecek apakah nilai x lebih besar dari 5. Jika kondisi pada if tidak terpenuhi, maka blok kode di dalam else akan dieksekusi.

else if

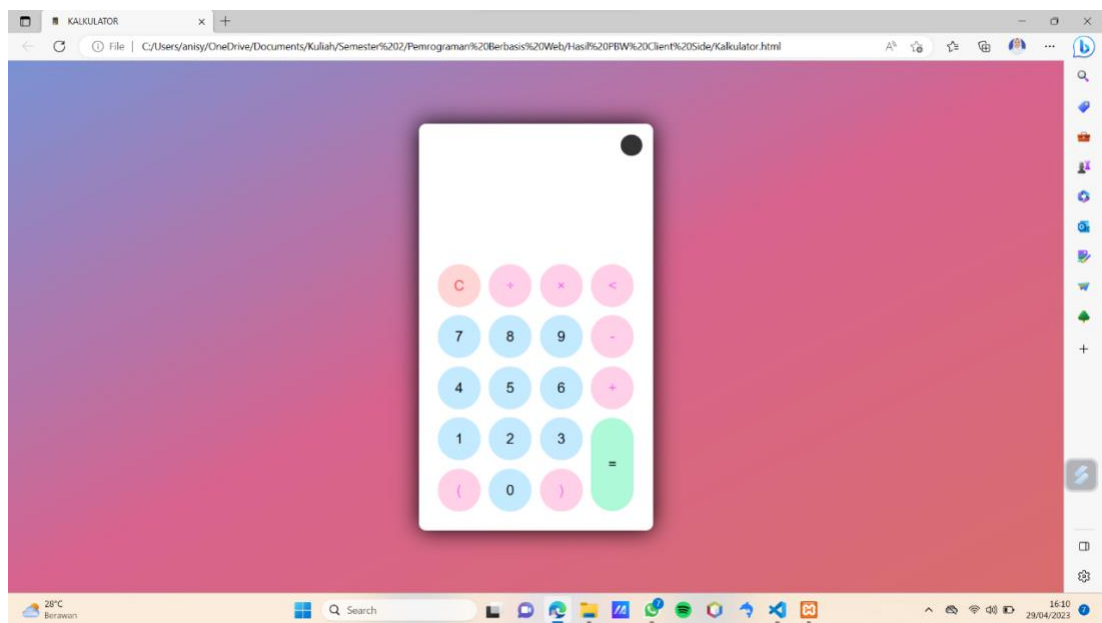
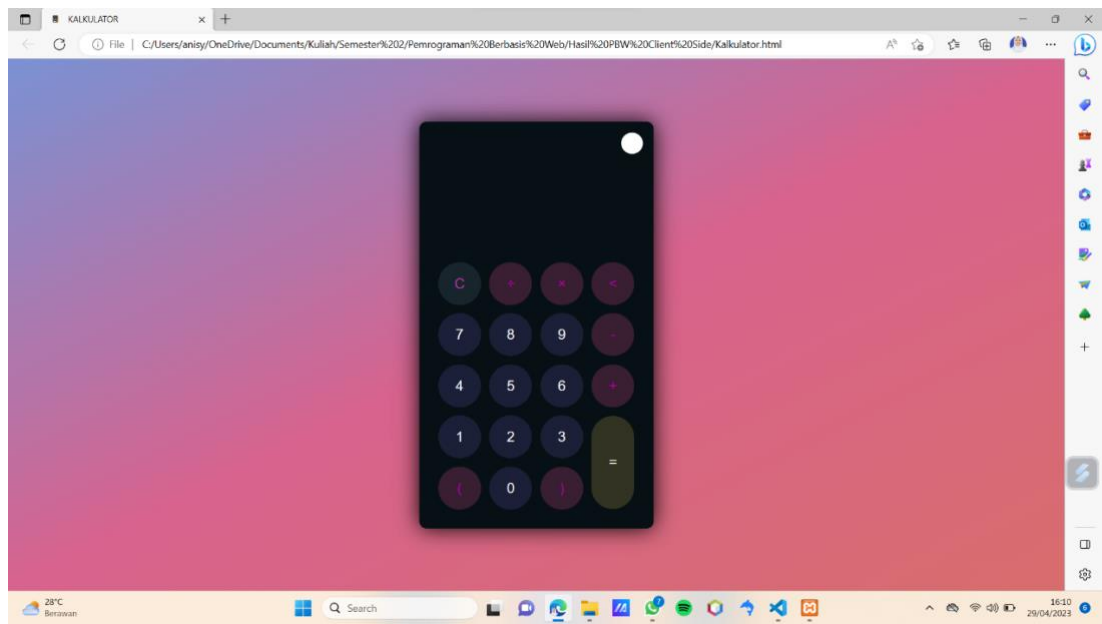
else if adalah kata kunci yang digunakan untuk mengeksekusi blok kode jika beberapa kondisi sebelumnya tidak terpenuhi. Berikut adalah contoh penggunaan if-else if:

javascript

Copy code

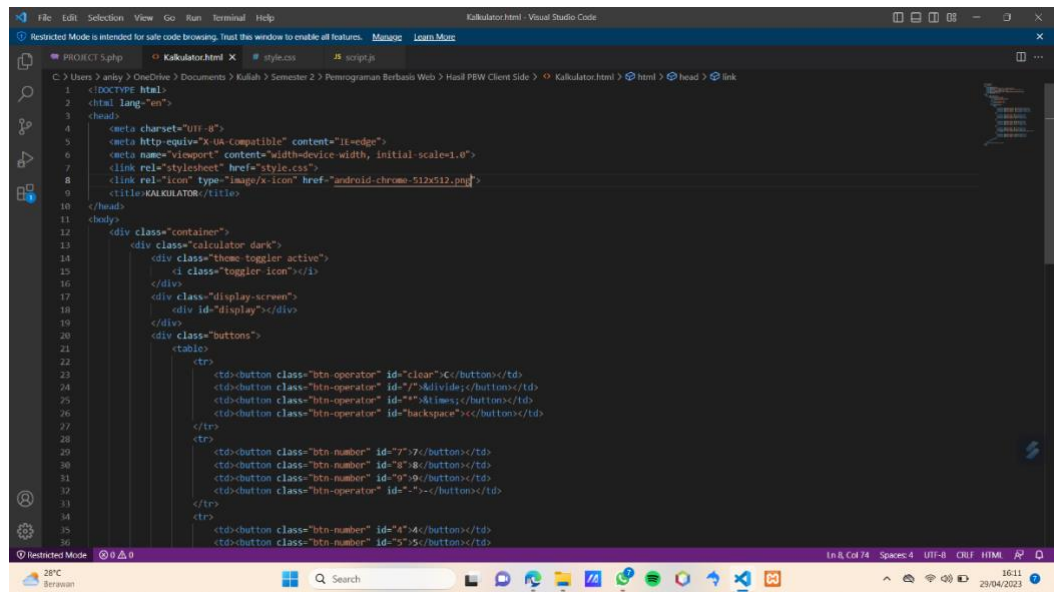
```
let x = 3;  
if (x > 5) {  
  console.log("x lebih besar dari 5");  
} else if (x > 2) {  
  console.log("x lebih besar dari 2 dan kurang dari atau sama dengan 5");  
} else {  
  console.log("x kurang dari atau sama dengan 2");  
}
```

2. Hasil

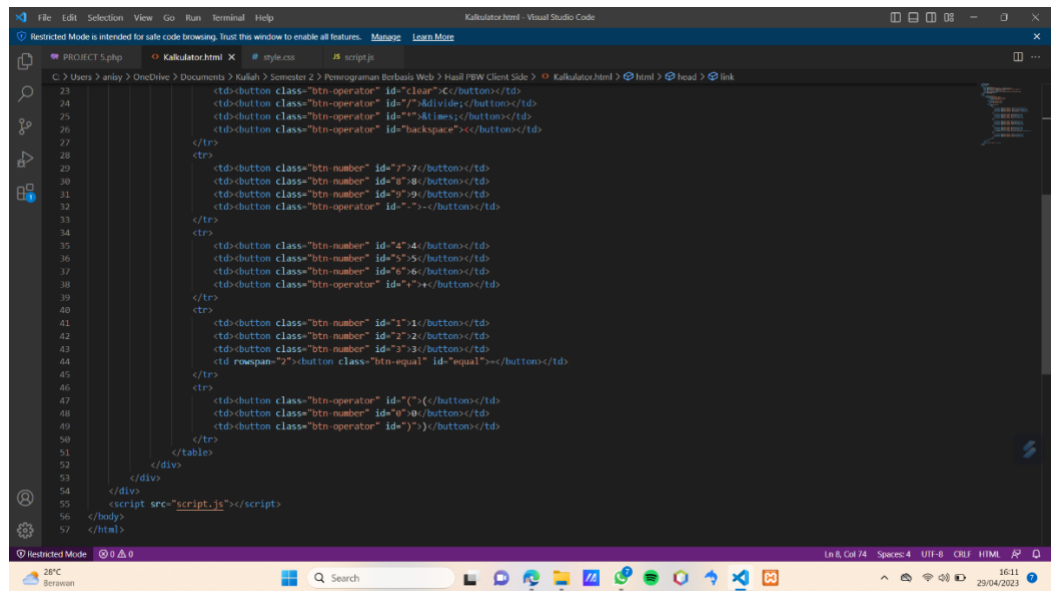


3. Source Code

➤ HTML



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <link rel="stylesheet" href="style.css">
8   <link rel="icon" type="image/x-icon" href="android-chrome-512x512.png">
9   <title>KALKULATOR</title>
10 </head>
11 <body>
12   <div class="container">
13     <div class="calculator dark">
14       <div class="theme-toggler active">
15         <i class="toggler icon"></i>
16       </div>
17       <div class="display screen">
18         <div id="display"></div>
19       </div>
20       <div class="buttons">
21         <table>
22           <tr>
23             <td><button class="btn-operator" id="clear">C</button></td>
24             <td><button class="btn-operator" id="/">÷</button></td>
25             <td><button class="btn-operator" id="*">×</button></td>
26             <td><button class="btn-operator" id="backspace"></button></td>
27           </tr>
28           <tr>
29             <td><button class="btn-number" id="7">7</button></td>
30             <td><button class="btn-number" id="8">8</button></td>
31             <td><button class="btn-number" id="9">9</button></td>
32             <td><button class="btn-operator" id="-">-</button></td>
33           </tr>
34           <tr>
35             <td><button class="btn-number" id="4">4</button></td>
36             <td><button class="btn-number" id="5">5</button></td>
```



```
37             <td><button class="btn-number" id="6">6</button></td>
38             <td><button class="btn-operator" id="+">+</button></td>
39           </tr>
40           <tr>
41             <td><button class="btn-number" id="1">1</button></td>
42             <td><button class="btn-number" id="2">2</button></td>
43             <td><button class="btn-number" id="3">3</button></td>
44             <td><button class="btn-operator" id="=">=</button></td>
45           </tr>
46           <tr>
47             <td><button class="btn-operator" id="(">(</button></td>
48             <td><button class="btn-number" id="0">0</button></td>
49             <td><button class="btn-operator" id=")">)</button></td>
50           </tr>
51         </table>
52       </div>
53     </div>
54   </div>
55   <script src="script.js"></script>
56 </body>
57 </html>
```

➤ CSS

The screenshot shows the Visual Studio Code editor with a CSS file named 'style.css'. The code defines styles for a calculator button and its container. The button has a padding of 0, margin of 0, box sizing of border-box, outline of 0, and a transition of all 0.5s ease. The body has a font family of sans-serif. A class 'a' has a text-decoration of none and a color of #fff. The body also has a background-image of a linear-gradient(to bottom right, #7a2d4d, #8d638e, #d963d9). The container has a height of 100vh, width of 100vw, display of grid, and place-items of center. The calculator has a position of relative, height of auto, width of auto, padding of 20px, border-radius of 10px, and a box-shadow of 0 0 30px #000.

```
1 {
2   padding: 0;
3   margin: 0;
4   box-sizing: border-box;
5   outline: 0;
6   transition: all 0.5s ease;
7 }
8
9 body {
10  font-family: sans-serif;
11 }
12
13 .a {
14  text-decoration: none;
15  color: #fff;
16 }
17
18 body {
19  background-image: linear-gradient(to bottom right, #7a2d4d, #8d638e, #d963d9);
20 }
21
22 .container {
23  height: 100vh;
24  width: 100vw;
25  display: grid;
26  place-items: center;
27 }
28
29 .calculator {
30  position: relative;
31  height: auto;
32  width: auto;
33  padding: 20px;
34  border-radius: 10px;
35  box-shadow: 0 0 30px #000;
36 }
```

The screenshot shows the Visual Studio Code editor with a CSS file named 'style.css'. The code defines styles for a theme toggler and a display. The theme-toggler has a position of absolute, top of 30px, right of 30px, color of #fff, cursor of pointer, and z-index of 1. The theme-toggler.active has a color of #333. The theme-toggler.active:before has a background-color of #fff. The theme-toggler:before has a content of '', height of 30px, width of 30px, position of absolute, top of 50%, transform of translate(50%, 50%), border-radius of 50%, background-color of #333, and z-index of -1. The display has a margin of 0 30px, height of 150px, width of auto, max-width of 200px, display of flex, and align-items of flex-end.

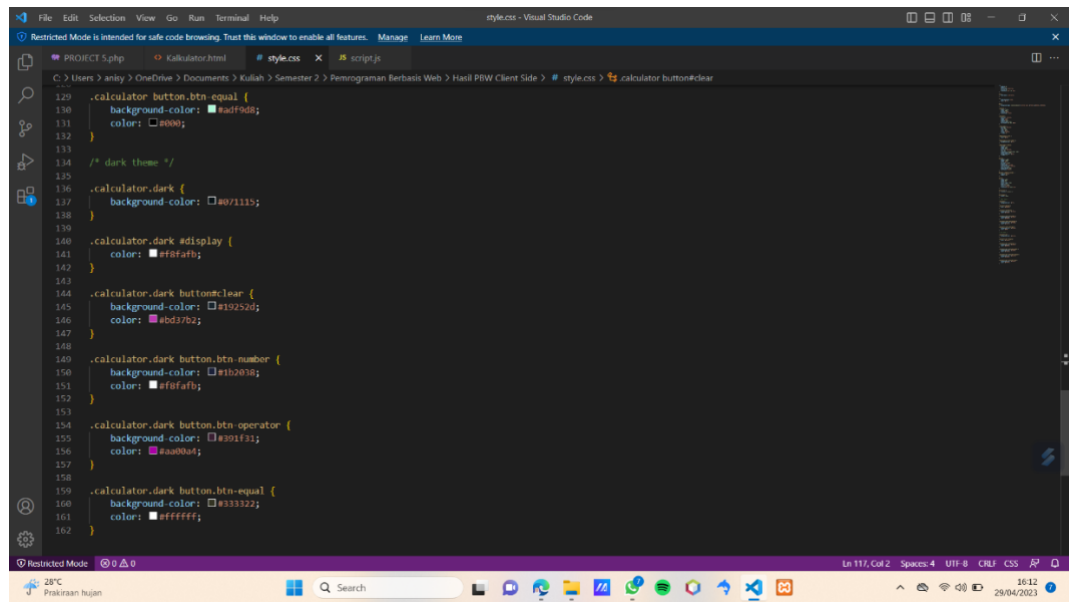
```
38 .theme-toggler {
39   position: absolute;
40   top: 30px;
41   right: 30px;
42   color: #fff;
43   cursor: pointer;
44   z-index: 1;
45 }
46
47 .theme-toggler.active {
48   color: #333;
49 }
50
51 .theme-toggler.active:before {
52   background-color: #fff;
53 }
54
55 .theme-toggler:before {
56   content: '';
57   height: 30px;
58   width: 30px;
59   position: absolute;
60   top: 50%;
61   transform: translate(50%, 50%);
62   border-radius: 50%;
63   background-color: #333;
64   z-index: -1;
65 }
66
67 .display {
68   margin: 0 30px;
69   height: 150px;
70   width: auto;
71   max-width: 200px;
72   display: flex;
73   align-items: flex-end;
74 }
```

```
File Edit Selection View Go Run Terminal Help
style.css - Visual Studio Code
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

PROJECT 5.php Kalkulator.html style.css script.js
C:\Users> anjay> OneDrive> Documents> Kuliah> Semester 2> Pemrograman Berbasis Web> Hasil PBO Client Side> style.css> calculator button#clear
67 #display {
68     margin: 0 10px;
69     height: 150px;
70     width: auto;
71     max-width: 270px;
72     display: flex;
73     align-items: flex-end;
74     justify-content: flex-end;
75     font-size: 30px;
76     margin-bottom: 20px;
77     overflow-x: scroll;
78 }
79
80 #display::webkit-scrollbar {
81     display: block;
82     height: 3px;
83 }
84
85 button {
86     height: 60px;
87     width: 60px;
88     border: 0;
89     border-radius: 30px;
90     margin: 5px;
91     font-size: 20px;
92     cursor: pointer;
93     transition: all 200ms ease;
94 }
95
96 button:hover {
97     transform: scale(1.1);
98 }
99
100 button:equal {
101     height: 130px;
102 }
```

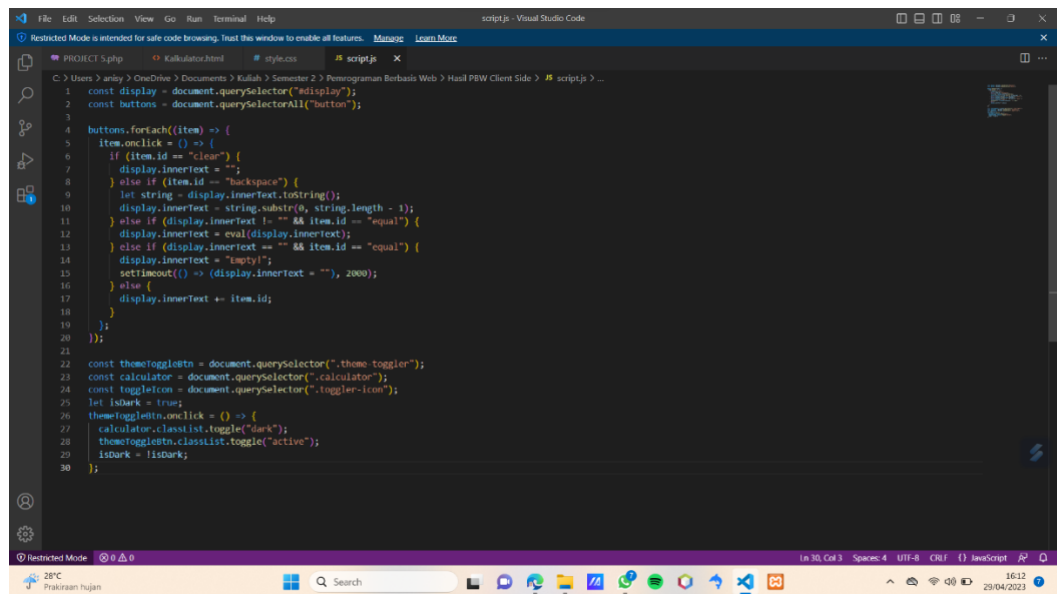
```
File Edit Selection View Go Run Terminal Help
style.css - Visual Studio Code
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

PROJECT 5.php Kalkulator.html style.css script.js
C:\Users> anjay> OneDrive> Documents> Kuliah> Semester 2> Pemrograman Berbasis Web> Hasil PBO Client Side> style.css> calculator button#clear
101     height: 130px;
102 }
103
104 /* light theme */
105
106 .calculator {
107     background-color: #fff;
108 }
109
110 .calculator #display {
111     color: #000000;
112 }
113
114 .calculator button#clear {
115     background-color: #ffdb5d;
116     color: #f44336;
117 }
118
119 .calculator button.btn-number {
120     background-color: #e3e3e3;
121     color: #000000;
122 }
123
124 .calculator button.btn-operator {
125     background-color: #ffdb5d;
126     color: #f44336;
127 }
128
129 .calculator button.btn-equal {
130     background-color: #e3e3e3;
131     color: #000000;
132 }
133
134 /* dark theme */
135
136 .calculator.dark {
```



```
129 .calculator button.btn-equal {
130   background-color: #eaf9d8;
131   color: #0000;
132 }
133
134 /* dark theme */
135
136 .calculator.dark {
137   background-color: #a01115;
138 }
139
140 .calculator.dark #display {
141   color: #f8fab;
142 }
143
144 .calculator.dark button.clear {
145   background-color: #19252d;
146   color: #bd37b2;
147 }
148
149 .calculator.dark button.btn-number {
150   background-color: #1b2018;
151   color: #f8fab;
152 }
153
154 .calculator.dark button.btn-operator {
155   background-color: #301f32;
156   color: #a000a0;
157 }
158
159 .calculator.dark button.btn-equal {
160   background-color: #333322;
161   color: #ffffff;
162 }
```

➤ JavaScript



```
1 const display = document.querySelector("#display");
2 const buttons = document.querySelectorAll("button");
3
4 buttons.forEach(item => {
5   item.onclick = () => {
6     if (item.id == "clear") {
7       display.innerText = "";
8     } else if (item.id == "backspace") {
9       let string = display.innerText.toString();
10      display.innerText = string.substr(0, string.length - 1);
11     } else if (display.innerText != "" && item.id == "equal") {
12       display.innerText = eval(display.innerText);
13     } else if (display.innerText == "" && item.id == "equal") {
14       display.innerText = "Error!";
15       setTimeout(() => (display.innerText = ""), 2000);
16     } else {
17       display.innerText += item.id;
18     }
19   };
20 });
21
22 const themeToggleBtn = document.querySelector("#theme-toggler");
23 const calculator = document.querySelector(".calculator");
24 const toggleIcon = document.querySelector("#toggle-icon");
25 let isDark = true;
26 themeToggleBtn.onclick = () => {
27   calculator.classList.toggle("dark");
28   themeToggleIcon.classList.toggle("active");
29   isDark = !isDark;
30 };
```