JavaScript Display Possibilities

JavaScript can "display" data in different ways:

- Writing into an HTML element, using innerHTML.
- Writing into the HTML output using document.write().
- Writing into an alert box, using window.alert().
- Writing into the browser console, using console.log().

Using innerHTML

To access an HTML element, JavaScript can use the document.getElementById(id) method. The id attribute defines the HTML element. The innerHTML property defines the HTML content:

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My First Paragraph</p>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = 5 + 6;
</script>

</body>
</html>
```

Using document.write()

For testing purposes, it is convenient to use document.write():

```
<!DOCTYPE html>
<html>
<body>
```

```
<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
document.write(5 + 6);
</script>

</body>
</html>
```

Using document.write() after an HTML document is loaded, will delete all existing HTML:

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<button type="button" onclick="document.write(5 + 6)">Try it</button>

</body>
</html>
```

Using window.alert()

You can use an alert box to display data:

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
window.alert("welcome to my page");
</script>

</body>
</html>
```

Using console.log ()

For debugging purposes, you can call the console.log() method in the browser to display data.

```
<!DOCTYPE html>
<html>
<body>

<script>
console.log (5 + 6);
</script>

</body>
</html>
```

JavaScript Print

JavaScript does not have any print object or print methods. You cannot access output devices from JavaScript. The only exception is that you can call the window.print() method in the browser to print the content of the current window.

In HTML, JavaScript code is inserted between <script> and </script> tags.

```
<!DOCTYPE html>
<html>
<body>

<button onclick="window.print()">Print this page</button>

</body>
</html>
```

Variables

Variables are Containers for Storing Data; JavaScript Variables can be declared in 4 ways:

- Automatically
- Using var

- Using let
- Using const

In this first example, x, y, and z are undeclared variables. They are automatically declared when first used:

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Variables</h1>

<p>In this example, x, y, and z are undeclared.</p>
<p>They are automatically declared when first used.</p>

<p id="demo"></p>

<script>
x = 5;
y = 6;
z = x + y;
document.getElementById("demo").innerHTML = "The value of z is: "+z;
</script>

</body>
</html>
```

Example using var

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Variables</h1>

<p>In this example, x, y, and z are variables.</p>

<p id="demo"></p>

<script>
var x = 5;
var y = 6;
var z = x + y;
```

```
document.getElementById("demo").innerHTML =
"The value of z is: " + z;
</script>

</body>
</html>
```

Example using let

```
<script>
let x = 5;
let y = 6;
let z = x + y;
document.getElementById("demo").innerHTML =
"The value of z is: " + z;
</script>
```

Example using const

```
<script>
const x = 5;
const y = 6;
const z = x + y;
document.getElementById("demo").innerHTML =
"The value of z is: " + z;
</script>
```

Mixed Example

```
<script>
const price1 = 5;
const price2 = 6;
let total = price1 + price2;
document.getElementById("demo").innerHTML =
"The total is: " + total;
</script>
```

The two variables **price1** and **price2** are declared with the const keyword. These are constant values and cannot be changed. The variable **total** is declared with the let keyword. The value total can be changed.

When to Use var, let, or const?

1. Always declare variables

2. Always use const if the value should not be changed

3. Always use const if the type should not be changed (Arrays and Objects)

4. Only use let if you can't use const

5. Only use var if you MUST support old browsers.

JavaScript Identifiers

All JavaScript **variables** must be **identified** with **unique names**. These unique names are called **identifiers**. Identifiers can be short names (like x and y) or more descriptive names (age, sum, totalVolume). The general rules for constructing names for variables (unique identifiers) are:

- Names can contain letters, digits, underscores, and dollar signs.
- Names must begin with a letter.
- Names can also begin with $.
- Names are case sensitive (y and Y are different variables).
- Reserved words (like JavaScript keywords) cannot be used as names.