JavaScript

JavaScript Introduction

JavaScript is a **programming language of the web**. It is used to make web pages **interactive and dynamic**.
Along with HTML and CSS, JavaScript is one of the **core technologies of web development**.

What Can JavaScript Do?

JavaScript can:

- Calculate and process data
- Manipulate and validate user input
- Change HTML content dynamically
- Modify CSS styles
- Respond to user actions like clicks and key presses
- Show or hide elements on a webpage

JavaScript runs directly in the **browser**, without needing any special software.

JavaScript Can Change HTML Content

JavaScript can access HTML elements and change their content using the getElementById() method.

Example

```
document.getElementById("demo").innerHTML = "Hello JavaScript";
```

JavaScript accepts **both single and double quotes**:

```
document.getElementById('demo').innerHTML = 'Hello JavaScript';
```

Use:
Used to update text, messages, results, or dynamic content on a web page.

JavaScript Can Change HTML Attribute Values

JavaScript can change HTML attributes like src, href, value, etc.

Example

```
document.getElementById("bulb").src = "bulbon.png";
```

Use:
Commonly used for image switching, form controls, and interactive UI elements.

JavaScript Can Change HTML Styles (CSS)

Changing CSS styles through JavaScript is another way of modifying HTML attributes.

Example

```
document.getElementById("demo").style.fontSize = "35px";
```

Use:
Used for animations, highlighting text, theme changes, and visual effects.

JavaScript Can Hide HTML Elements

HTML elements can be hidden by changing the display property.

Example

```
document.getElementById("demo").style.display = "none";
```

Use:
Used in popups, menus, modals, and conditional content display.

JavaScript Can Show HTML Elements

Hidden elements can be shown again by changing the display property.

Example

document.getElementById("demo").style.display = "block";

Use:
Used for showing messages, forms, or content on user actions.

JavaScript Where To

The <script> Tag

JavaScript code is written inside the <script> and </script> tags.

Example

```
<script>

document.getElementById("demo").innerHTML = "My First JavaScript";

</script>
```

The type="text/javascript" attribute is **not required**, because JavaScript is the default scripting language in HTML.

JavaScript Functions and Events

A **JavaScript function** is a block of code that runs when it is called.

Functions are often triggered by **events** such as:

- Button click
- Mouse hover
- Page load

Example

```
<button onclick="myFunction()">Click Me</button>
```

JavaScript in <head>

JavaScript can be placed inside the <head> section.

Example

```
<!DOCTYPE html>

<html>

<head>

<script>

function myFunction() {

  document.getElementById("demo").innerHTML = "Paragraph changed.";

}

</script>

</head>

<body>


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

<button onclick="myFunction()">Try it</button>


</body>

</html>
```

Use:
Good for functions that are used across the page.

JavaScript in <body>

JavaScript can also be placed inside the <body> section.

Example

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

<p id="demo">A Paragraph</p>
<button onclick="myFunction()">Try it</button>

<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>

</body>
</html>
```

Advantage:
Placing scripts at the bottom of <body> improves page loading speed.

External JavaScript

JavaScript code can be written in a separate file with .js extension.

External File: myScript.js

```
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
```

Linking External JavaScript

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

Use:
Best for large projects and reusable code.

Advantages of External JavaScript

- Separates HTML and JavaScript
- Makes code cleaner and easier to maintain
- Improves performance using browser caching
- Same file can be used in multiple pages

Multiple files can be added like this:

```
<script src="myScript1.js"></script>

<script src="myScript2.js"></script>
```

External JavaScript References

External JavaScript files can be linked in three ways:

1. **Full URL**

```
<script src="https://www.w3schools.com/js/myScript.js"></script>
```

2. **File Path**

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

3. **Without Path**

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

JavaScript Output

JavaScript can display (output) data in **different ways**, depending on **where** and **why** you want to show the result.

JavaScript Display Possibilities

JavaScript can display data by:

1. **Writing into an HTML element** (innerHTML / innerText)
2. **Writing directly to the page** (document.write)
3. **Showing a popup message** (alert)
4. **Printing to the browser console** (console.log)

---

1. Using innerHTML

What it does

- Writes **HTML content** inside an HTML element.
- You can add **tags**, styling, headings, etc.

How it works

- First, select an element using getElementById()
- Then change its content using innerHTML

Example

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

<h1>My First Web Page</h1>
<p id="demo"></p>
```

```
<script>
document.getElementById("demo").innerHTML = "<h2>Hello World</h2>";
</script>

</body>
</html>
```

Output

Displays **Hello World** as an <h2> heading.

Use

Most commonly used

When you want to insert **HTML + text**

---

2. Using innerText

What it does

- Writes **only plain text**
- HTML tags are treated as text, not code

Example

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

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

<script>
document.getElementById("demo").innerText = "Hello World";
```

```
</script>


</body>

</html>
```

Difference from innerHTML

innerHTML	**innerText**
Can use HTML tags	Only text
<h1> works	<h1> shows as text

Use

When you only want to change **text**, not HTML

---

3. Using document.write()

What it does

- Writes content **directly to the web page**

Example

```
<script>

document.write(5 + 6);

</script>
```

Example

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

Clicking the button removes everything and shows 11.

Use

Not recommended
Only for **testing or learning**

---

4. Using window.alert()

What it does

- Shows a **popup alert box**

Example

```
<script>
window.alert(5 + 6);
</script>
```

You can also write:

```
alert(5 + 6);
```

Why window is optional

- window is the **global object** in JavaScript
- So alert() automatically belongs to window

---

5. Using console.log()

What it does

- Prints output in the **browser console**
- Used mainly for **debugging**

Example

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

Where to see output

Open **Inspect → Console**

Use

Best for developers
Debugging errors and values
User cannot see it

---

6. JavaScript Print (window.print())

Important Point

JavaScript **cannot directly control printers**.

Only available option

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

What it does

- Opens the browser's **print dialog**
- Prints the **current webpage**

Use

Printing invoices, forms, reports

Quick Summary Table

Method	**Purpose**	**Visible to User**
innerHTML	Display HTML + text	Yes
innerText	Display only text	Yes
document.write	Write to page (testing)	Yes
alert	Popup message	Yes
console.log	Debugging	No
window.print	Print page	Yes
