# JavaScript in the Browser

---

## Semicolons and Whitespace

- **Semicolons (`;`) are optional** in JavaScript.
- JavaScript **ignores spaces, tabs, and newlines**, unless inside strings.
- Originally created to **add interactivity** to web pages.
- Runs inside the browser using a **JavaScript Engine** (like V8 in Chrome).

🔒 **Limitation**: JavaScript's access is limited to ensure **browser security**.

---

## 🛠️ Developer Tools in Browsers

Every modern browser (Chrome, Firefox, Edge, etc.) includes built-in **Developer Tools**:

- **Elements Tab** 🧱 – View and inspect HTML/CSS elements.
- **Console Tab** 📜 – View logs, errors, and debug JavaScript.
- **Network Tab** 🌐 – Inspect network requests.

Open DevTools with:

- `Ctrl + Shift + I` or `F12`

---

## 🧠 Ways to Use JavaScript in the Browser

### 1️⃣ Inline Script (Embedded in HTML)

```
<script>
  console.log("Hello from inline script!");
</script>
```

### 2️⃣ External Script (Recommended ✅)

```
<script src="script.js"></script> <!-- Place just before </body> -->
```

#### 🔄 Why use external JS?

- Better organization (Separation of Concerns)
- Browser caching makes your site faster 🚀

---

## 💡 JavaScript in Action (Example)

```html
<!DOCTYPE html>
<html>
  <head>
    <title>JS Example</title>
  </head>
  <body>
    <h1 id="heading">Hello, World!</h1>
    <button id="changeBtn">Change Text</button>

    <script>
      const heading = document.getElementById('heading');
      const changeBtn = document.getElementById('changeBtn');

      changeBtn.addEventListener('click', function () {
        heading.textContent = 'Text Changed! ✨';
      });
    </script>
  </body>
</html>
```

### 🧪 Output:

Clicking the **"Change Text"** button updates the heading to "Text Changed! ✨"

---

## 🧾 The `console` Object & Its Methods

The `console` object helps developers **log, debug, and inspect** code in the browser.

| Method | Description | Example / Output |
|---|---|---|
| `console.log()` | Log regular message ▨ | `"Hello, World!"` |
| `console.error()` | Log error ❌ | `"An error occurred."` |
| `console.warn()` | Log warning ⚠️ | `"This is a warning."` |
| `console.info()` | Informational message ℹ️ | `"Info log"` |
| `console.debug()` | Debugging message 🐞 | `"Debug info"` |
| `console.assert()` | Shows error if condition is false | `Value should be > 20` |
| `console.clear()` | Clears the console 🧹 | – |
| `console.table()` | Tabular view of data 📋 | [Table Output] |
| `console.count()` | Logs call count by label | Counter: 1, 2... |
| `console.time()` / `console.timeEnd()` | Measure performance/time ⏱️ | Timer: X ms |

### 🧪 `console.assert()` Example:

```
const value = 10;
console.assert(value > 20, '❌ Value should be greater than 20');
```

### 🧪 `console.table()` Example:

```
const users = [
  { name: "John", age: 25 },
  { name: "Jane", age: 30 }
];
console.table(users);
```

---

## ⏱️ Performance: `for` vs `while` Loop Comparison

```
console.time("forLoop");
for (let i = 0; i < 500; i++) {
  console.log(233);
}
console.timeEnd("forLoop");

console.time("whileLoop");
let i = 0;
while (i < 500) {
  console.log(233);
  i++;
}
console.timeEnd("whileLoop");
```

### ⏱️ Output:

```
forLoop: 100ms
whileLoop: 98ms
```

*(Example times, may vary)*

---

## 🧰 Full List of `console` Methods

| Method | Description |
|---|---|
| `assert()` | Shows error if assertion fails |
| `clear()` | Clears console |

| Method | Description |
|---|---|
| `count()` | Counts label usage |
| `error()` | Error message |
| `group()` / `groupEnd()` | Indented grouping of logs |
| `groupCollapsed()` | Collapsed log group |
| `info()` | Informational log |
| `log()` | Normal log |
| `table()` | Data as table |
| `time()` / `timeEnd()` | Track time |
| `trace()` | Stack trace |
| `warn()` | Warning |

## 🔄 User Interaction: `prompt`, `alert`, `confirm`

| Function | Description | Example |
|---|---|---|
| `prompt()` | Takes input from user 🧑‍💻 | `prompt("Enter your name")` |
| `alert()` | Shows message 📢 | `alert("Hello!")` |
| `confirm()` | Ask for confirmation ❓ | `confirm("Are you sure?")` |

Example:

```
const name = prompt("What’s your name?");
alert("Hi, " + name + "!");

const confirmDelete = confirm("Are you sure?");
if (confirmDelete) {
  console.log("Item deleted.");
} else {
  console.log("Cancelled.");
}
```

## 🌐 The Window, DOM, and BOM

```
+-----------------------+
|       Window          | 🌐 Global object
+-----------------------+
         |
         | access to:
         ↓
+-----------------------+
|     DOM (Document)     | 📄 HTML structure as JS object
+-----------------------+
         |
         ↓
+-----------------------+
|     BOM (Browser)      | 🧭 Browser-related info (URL, screen, etc.)
+-----------------------+
```

### ✨ DOM (Document Object Model)

- Represents **HTML page structure** as a tree.
- Access with:
  - `document.getElementById()`
  - `document.querySelector()`
- Change properties like: `document.body.style.background = "green"`

### ✨ BOM (Browser Object Model)

- Interact with **browser features**:
  - `window.alert()`, `window.location`, `window.history`
  - `window.open()`

```
window.location.href = "https://dpvasani56.com";
```

---

## 📝 Practice Questions (with Answers)

> 📚 From Ch.6 – JavaScript in Browser

1️⃣ **Log message**:

```
console.log("Hello, World!");
```

2️⃣ **Error message**:

```
console.error("An error occurred.");
```

3️⃣ **Clear console**:

```
console.clear();
```

4️⃣ **Warning message**:

```
console.warn("This is a warning.");
```

5️⃣ **Tabular data**:

```
const data = [
  { name: "John", age: 25 },
  { name: "Jane", age: 30 }
];
console.table(data);
```

6️⃣ **Assertion (if condition is false)**:

```
const value = 10;
console.assert(value > 20, "Value should be greater than 20");
```

---