

✓ defer

html

CopyEdit

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

What it does:

- Downloads the JS file in parallel with HTML parsing.
- Waits to execute the JS until the HTML is fully parsed (DOMContentLoaded).
- Maintains the order of scripts if multiple have defer.

Use when:

- You want scripts to run after the page is loaded.
- The script depends on DOM elements being present.

✓ async

html

CopyEdit

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

What it does:

- Downloads the JS file in parallel with HTML parsing.
- Executes the JS as soon as it's downloaded — does not wait for HTML to finish parsing.
- Scripts may execute out of order if multiple have async.

Use when:

- The script is independent (e.g. analytics, ads, etc.).
- It doesn't rely on the DOM or other scripts.

```
2 let heading=document.querySelectorAll('h1') // returns nodelist
3 let heading2=document.getElementsByTagName('h1') // returns HTMLCOLLECTION
4 console.log(heading)
```

EXPLORER

JS WORK

Practice

index.htmlU

JS practice.jsU

Practice > JS practice.js > ...

```
1 console.log("HELLO");
2 let heading=document.querySelectorAll('h1') // returns nodelist
3 // heading.style.background='red'
4
5 for (let elem of heading) {
6   console.log(elem.innerHTML)
7 }
```

innerHTML

👉 Gets or sets the HTML content (including tags) inside an element.

```
html
<div id="demo"><b>Hello</b> World</div>
```

```
js
let e1 = document.getElementById('demo');
console.log(e1.innerHTML); // "<b>Hello</b> World"
```

✅ You can insert HTML elements using **innerHTML**:

```
js
e1.innerHTML = "<i>Hi</i> there"; // Now the div contains italic text
```

⚠ Security risk if you're inserting untrusted content (can lead to XSS).

textContent

👉 Gets or sets only the text (no HTML tags).

```
html
<div id="demo"><b>Hello</b> World</div>
```

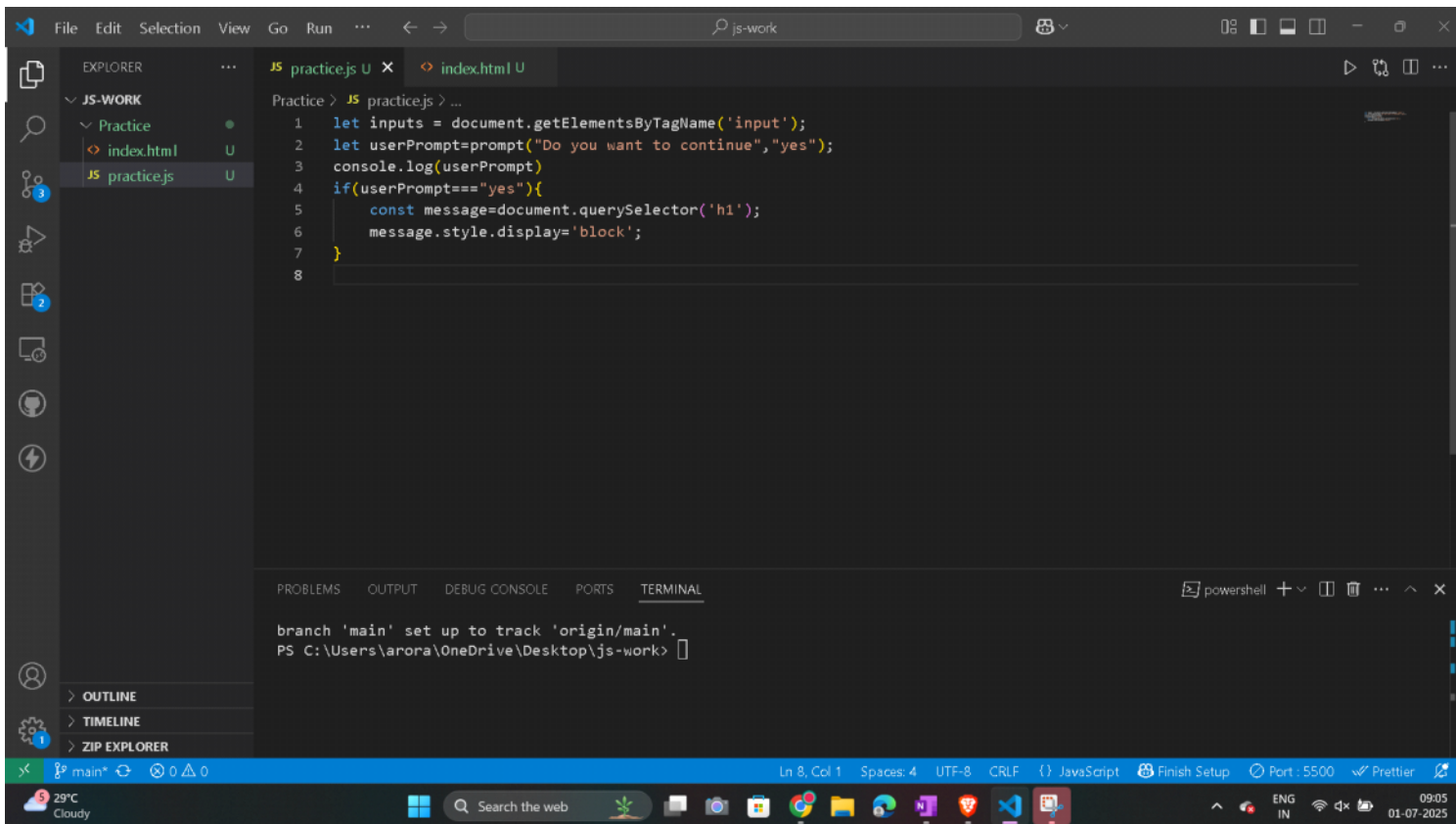
```
js
let e1 = document.getElementById('demo');
console.log(e1.textContent); // "Hello World"
```

✅ Use it when you just want plain text.

```
js
e1.textContent = "<i>Hi</i> there"; // Displays: <i>Hi</i> there (as plain text)
```

Playard

Visual Studio Code interface showing a web project named "js-work". The Explorer sidebar shows the file structure: "index.html" and "practice.js". The main editor displays the HTML content of "index.html", which includes a DOCTYPE declaration, meta tags for charset and viewport, a title "Document", and a script tag for "practice.js". The body contains a hidden h1 element with the text "WELCOME". The bottom status bar shows "main" branch, 0 changes, and various settings like "Ln 12, Col 8", "Spaces: 4", "UTF-8", "CRLF", and "HTML".



what is es5 and what is new

Great question!

◆ What is **ES5**?

ES5 stands for ECMAScript 5, a version of the JavaScript language standard released in 2009. It was the most widely used version for many years and is supported by all browsers (even IE9+).

Manan2505/JavaScript | javascript documentation - Yai | JavaScript Guide - JavaScript | Searching: getElement, query | Best JS Documentation

chatgpt.com/q/6863499c-c7ac-800c-bb00-1b044c6e27fa

ChatGPT

What is Newer Than ES5? (Mostly ES6/ES2015 and beyond)

ES6 (ECMAScript 2015) introduced major changes and is the foundation of modern JavaScript. Here's what's new:

ES6+ Features (Post-ES5)

Feature	Description	Example
let and const	Block-scoped variables	let x = 10; const y = 5;
Arrow functions	Shorter function syntax	(a, b) => a + b
Template literals	Backtick strings with <code>`\${}`</code>	<code>`hello, \${name}`</code>
Destructuring	Extract from objects/arrays	let {a, b} = {1, 2}
Default parameters	Set default in function args	function greet(name = 'Guest')
Rest & Spread	<code>...args</code> , <code>...arr</code> for flexibility	function(...args)
Classes	OOP-style syntax	class Person { constructor() {} }
Modules	import / export	import x from './file.js'
Promises	Async handling	fetch().then().catch()
for...of loop	Iterate over iterable objects	for (let item of array)
Map, Set	New data structures	let map = new Map()
Async/Await (ES2017)	Cleaner async code	async function load() {}

Ask anything

+ Tools

ChatGPT can make mistakes. Check important info. See [Cookie Preferences](#).

29°C Cloudy

Search the web

ENG IN 09:10 01-07-2025

1. var – Old and outdated (ES5)

- Function-scoped
- Hoisted and initialized as `undefined`
- Can be redeclared and updated
- Avoid in modern JS

```
js
var x = 10;
var x = 20; // allowed
```

```
js
function test() {
  if (true) {
    var a = 5;
  }
  console.log(a); // 5 (var is function-scoped)
}
```

2. let – Modern and preferred

- Block-scoped (`{ }`)
- Hoisted, but not initialized
- Can be updated, but not redeclared in the same scope

```
js
let y = 10;
y = 20; // allowed
let y = 30; // Error: already declared
```

```
js
function test() {
  if (true) {
    let b = 10;
  }
  console.log(b); // ReferenceError
}
```

3. `const` – Constant (unchangeable reference)

- ☒ Block-scoped
- ! Must be initialized when declared
- ✗ Cannot be updated or redeclared

```
js
const z = 100;
z = 200; // ✗ Error: assignment to constant variable
```

However:

```
js
const arr = [1, 2];
arr.push(3); // ☒ Allowed (array reference not changed)
```

☒ When to Use:

- Use `let` for variables that change.
- Use `const` for constants and references.
- Avoid `var` in modern code unless maintaining legacy code.

Let me know if you want visual examples or interview-style questions on this topic!

The screenshot shows a web browser window with a ChatGPT interface. The browser's address bar displays a URL from chatgpt.com. The ChatGPT interface has a sidebar on the left with icons for chat, search, and other features. The main content area is titled "Post/Pre-Increment Interview Questions (No Answers Yet)" and contains five numbered questions, each with a code snippet in JavaScript. The questions are:

- What will be the output of the following?

```
let x = 0;
console.log(x++);
console.log(x);
```
- What will be logged to the console?

```
let x = 10;
let y = ++x;
console.log(x, y);
```
- Consider this:

```
let a = 0;
let b = a++ + ++a;
console.log(a, b);
```
- What happens here?

```
let count = 1;
count = count++ + ++count;
console.log(count);
```
- Find the final value of `z`.

```
let z = 0;
z = z++ - --z + ++z - z--;
```

At the bottom of the ChatGPT interface, there is a text input field with the placeholder "Ask anything" and a "Tools" button. The browser's taskbar at the bottom shows the Windows logo, a search bar, and various application icons. The system tray on the right indicates the language is "ENG IN", the date is "01-07-2025", and the time is "09:38".

JavaScript Guide - JavaScript | M... x | Searching: getElement*, queryS... x | Best JS Documentation x +

chatgpt.com/q/6863499c-c7ac-800c-bb00-1b044de27fa

ChatGPT

count = count++ + ++count;
console.log(count);

5. Find the final value of `z`:

```
js
let z = 0;
z = z++ + --z + ++z - z--;
console.log(z);
```

6. Evaluate this expression:

```
js
let num = 0;
let result = num++ + ++num;
console.log(result);
```

7. What will be printed?

```
js
let i = 5;
console.log(++i + ++i);
```

8. Predict the output:

```
js
let x = 5;
x = ++x + x++;
console.log(x);
```

Let me know which question you want the answer to, and I'll explain it step by step.

Ask anything

ChatGPT can make mistakes. Check important info. See [Code Preferences](#).

30°C Lightning nearby

Search the web

ENG IN 09:38 01-07-2025

In Simple Terms:

TDZ is like a trap.

You declared the variable, but until JS *initializes* it, it stays in a "dead zone" and throws an error if touched.

Using `.className`

```
js
div.className = "my-class";
```

- ◆ Replaces all existing class names.

Using `.classList.add()` (Recommended)

```
js
div.classList.add("my-class");
```

- ◆ Adds a class without affecting others (safer for multiple classes)

JavaScript Guide - JavaScript | M | x | x | Searching: getElement*, queryS | x | Best JS Documentation | x | +

chatgpt.com/q/6863499c-c7ac-800c-bb00-1ba44de27fa

ChatGPT

Great question! Both `appendChild()` and `append()` are used to add elements to the DOM, but they have some key differences:

• `appendChild()`

- Appends only one Node (like an element or text node).
- Throws an error if you try to pass multiple nodes or plain strings.
- Returns the appended node.
- Cannot append text directly.

```
jsconst div = document.createElement('div');div.textContent = 'Hello';document.body.appendChild(div); // Worksdocument.body.appendChild('Hello'); // Errors: only nodes allowed
```

• `append()`

- Newer (ES6+), more flexible.
- Can append multiple nodes or strings.
- Does not return the appended element (returns `undefined`).
- Can append both DOM nodes and text.

```
jsconst div = document.createElement('div');div.textContent = 'Hello';document.body.append(div); // Worksdocument.body.append('World'); // Works
```

Ask anything

+ Tools

ChatGPT can make mistakes. Check important info. See Code Preferences.

30°C Cloudy

Search the web

ENG IN 09:53 01-07-2025