1. What is ES6?

ES6, also called **ECMAScript 2015**, is the **6th edition of the JavaScript standard**.

- JavaScript is based on **ECMAScript**, which defines how the language works.
- ES6 introduced **modern syntax and features** to make JavaScript **cleaner, more powerful, and easier to maintain**.
- Before ES6, JavaScript was functional but limited; ES6 brought features like **classes, modules, arrow functions, template literals, promises**, etc.

Think of it like "JavaScript 2.0" — it's still JavaScript, but with new, easier ways to code.

---

2. Key ES6 Features (With Explanation)

Feature	What It Does	Example
let / const	Block-scoped variables (safer than var)	`let x = 10; const y = 20;`
Arrow Functions	Shorter function syntax, no own `this`	`const add = (a,b) => a+b;`
Template Literals	Strings with variables & multi-line	`` `Hello ${name}` ``
Destructuring	Extract values from arrays/objects	`const {name, age} = person;`
Default Parameters	Set default values in functions	`function greet(name="Guest"){}`
Spread & Rest Operators	Expand or collect elements	`[...arr], function sum(...args)`
Classes	Object-oriented syntax	`class Person { constructor(name){ this.name=name; } }`
Modules	Import/export code across files	`export const add=()=>{}; import {add} from './file.js';`
Promises	Handle asynchronous tasks	`fetch(url).then(...).catch(...);`

Feature	What It Does	Example
Enhanced Object Literals	Shorthand for properties/methods	`{name, greet(){}}`

---

3. How to Run ES6 Code

You can run ES6 in **2 main ways**:

Option 1: In Browser (Quickest Way)

1. Open **any modern browser** (Chrome, Firefox, Edge).
2. Open **Developer Tools → Console** (Right-click → Inspect → Console).
3. Type your ES6 code and press **Enter**.

Example:

```
const greet = (name="Guest") => {
  console.log(`Hello, ${name}!`);
};

greet("Alice"); // Hello, Alice!
```

Option 2: Using a Text Editor (Saved File)

1. Open a **text editor** (VS Code, Sublime Text, Notepad++).
2. Create a new file: `example.js`.
3. Write ES6 code in that file.

example.js

```
// ES6 Example
const numbers = [1, 2, 3];
const doubled = numbers.map(n => n * 2);
console.log(doubled);
```

4. Open a browser and run it via **HTML file**, or use **Node.js**.

Option A: Run in Browser via HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>ES6 Example</title>
</head>
<body>
  <script src="example.js"></script>
</body>
</html>
```

- Save as `index.html`
- Open `index.html` in a browser → check Console (F12) → see output

Option B: Run with Node.js (No Browser Needed)

1. Install Node.js
2. Open terminal → navigate to folder
3. Run:

```
node example.js
```

4. Output will appear in terminal

---

4. Tips for Running ES6 Smoothly

- Use **modern browsers** (Chrome, Firefox, Edge) — they support ES6 fully.
- If using **Node.js**, use latest version (v12+) to support ES6 features.
- For older browsers, you can use **Babel** to transpile ES6 → ES5.

---

5. Quick Example Combining Features

```
// ES6 Full Example
class Person {
  constructor(name, age) {
    this.name = name;
    this.age = age;
  }

  greet() {
    console.log(`Hi, I'm ${this.name} and I'm ${this.age} years old.`);
  }
}

const alice = new Person("Alice", 25);
```

```
alice.greet();

const numbers = [1, 2, 3];
const doubled = numbers.map(n => n * 2);
console.log(doubled);
```

- Copy → save as `example.js` → run in browser or Node.js → works perfectly.

ES6 cheat sheet you can use for quick reference or teaching. I'll organize it in a clean, easy-to-digest format with **examples**.

JavaScript ES6 Cheat Sheet

1. `let` and `const`

```
let x = 10;       // block-scoped variable
x = 20;           // allowed

const y = 30;     // constant, cannot reassign
// y = 40;        // □ Error
```

---

2. Arrow Functions

```
// Traditional
function add(a, b) {
  return a + b;
}

// Arrow Function
const add = (a, b) => a + b;

// Single parameter
const square = n => n * n;
```

---

3. Template Literals

```
const name = "Alice";
const age = 25;

// Multi-line & variable embedding
const greeting = `Hello, ${name}!You are ${age} years old.`;
```

```
console.log(greeting);
```

---

4. Destructuring

```
// Array destructuring
const arr = [1, 2, 3];
const [a, b] = arr; // a=1, b=2

// Object destructuring
const person = {name: "Bob", age: 30};
const {name, age} = person; // name="Bob", age=30
```

---

5. Default Parameters

```
function greet(name = "Guest") {
  console.log(`Hello, ${name}`);
}

greet();          // Hello, Guest
greet("Alice");   // Hello, Alice
```

---

6. Spread & Rest Operators

```
// Spread - expand array
const nums = [1, 2, 3];
const newNums = [...nums, 4]; // [1,2,3,4]

// Rest - collect arguments
function sum(...args) {
  return args.reduce((a, b) => a + b, 0);
}
console.log(sum(1,2,3,4)); // 10
```

---

7. Classes

```
class Person {
  constructor(name, age) {
    this.name = name;
    this.age = age;
  }
  greet() {
    console.log(`Hi, I'm ${this.name}`);
  }
}

const bob = new Person("Bob", 30);
```

```
bob.greet(); // Hi, I'm Bob
```

---

8. Modules

```
// math.js
export const add = (a, b) => a + b;

// main.js
import { add } from './math.js';
console.log(add(2,3)); // 5
```

---

9. Promises

```
fetch('https://api.example.com/data')
  .then(response => response.json())
  .then(data => console.log(data))
  .catch(err => console.error(err));
```

---

10. Enhanced Object Literals

```
const name = "Alice";

const user = {
  name,         // shorthand property
  greet() {     // shorthand method
    console.log("Hi!");
  }
};

user.greet(); // Hi!
```
