Features of ES6

ES6 introduced many enhancements to JavaScript. Some major features include:

1. **let and const** – Block-scoped variable declarations
2. **Arrow functions** – Shorter syntax for function expressions
3. **Classes** – Cleaner syntax for object-oriented programming
4. **Template literals** – ${} interpolation inside backticks (``)
5. **Default parameters** – Function parameters with default values
6. **Destructuring** – Extract values from arrays/objects easily
7. **Rest & Spread operators** – ...args for collecting/spreading
8. **Modules (import/export)** – Code modularization
9. **Promises** – Handle asynchronous operations
10. **Map and Set** – New collection data structures

let in JavaScript

- let is used to declare **block-scoped variables**.
- Unlike var, it **does not get hoisted** to the top of the block.
- It **cannot be redeclared** in the same scope.

javascript

```
let x = 10;
if (true) {
  let x = 20; // different 'x'
  console.log(x); // 20
}
console.log(x); // 10
```

Differences between var and let

Feature	var	let
Scope	Function-scoped	Block-scoped ({})
Hoisting	Yes (initialized as undefined)	Yes (but not initialized)
Redeclaration	Allowed	Not allowed in same scope
Use in loops	May cause issues	Safer in loops

const in JavaScript

- const is used to declare **block-scoped constants**.
- **Must be initialized at declaration**.
- Cannot be reassigned, but **objects and arrays can be mutated**.

Javascript

```
const PI = 3.14;
// PI = 3.15;  Error


const person = { name: "Vaishnavi" };
person.name = "Sai"; // Allowed (mutating object)
```

ES6 Class Fundamentals

- Provides a clean, OOP-style syntax.
- Uses constructor to initialize objects.
- Methods are defined inside the class.

Javascript

```
class Person {
  constructor(name) {
    this.name = name;
  }
  greet() {
    console.log(`Hello, I am ${this.name}`);
  }
}
const p = new Person("Vaishnavi");
p.greet(); // Hello, I am Vaishnavi
```

ES6 Class Inheritance

- Use extends to inherit from another class.
- Use super() to call the parent class constructor.

javascript

```
class Student extends Person {
  constructor(name, roll) {
    super(name);
    this.roll = roll;
  }
  show() {
    console.log(`${this.name} - Roll No: ${this.roll}`);
  }
}
const s = new Student("Vaishnavi", 101);
s.show(); // Vaishnavi - Roll No: 101
```

Arrow Functions

- Shorter syntax for function expressions.
- this is **lexically bound** (no own this).
- No arguments object.

javascript

```
const add = (a, b) => a + b;
console.log(add(3, 4)); // 7

const greet = name => console.log(`Hi, ${name}`);
```

greet("Vaishnavi"); // Hi, Vaishnavi

Set and Map in ES6

◆ **Set**

- A collection of **unique values** (no duplicates).

javascript

CopyEdit

const set = new Set([1, 2, 3, 2]);

set.add(4);

console.log(set); // Set(4) {1, 2, 3, 4}

◆ **Map**

- A collection of **key-value pairs**.
- Keys can be of any type.

Javascript

const map = new Map();

map.set("name", "Vaishnavi");

map.set(1, "One");

console.log(map.get("name"));