What are Access Specifiers?

✅ Definition:

Access specifiers (also called access modifiers) define the **visibility** or **access level** of class members (properties or methods).They control **who can access what** from outside or inside the class.

✅ Access Specifiers in JavaScript (ES6+):

Specifier	Symbol	Meaning
public	*(default)*	Can be accessed from **anywhere**.
private	#	Can be accessed **only inside** the class.
protected	❌	*Not natively supported in JS(Can be simulated with _ naming convention)*

✅ Example in JavaScript:

```
class Person {
  // public property (default)
  name;

  // private property (using #)
  #ssn;

  constructor(name, ssn) {
    this.name = name;     // public
    this.#ssn = ssn;      // private
  }

  // public method
  getSSN() {
    return this.#ssn;
  }
}

const p1 = new Person("Milan", "123-45-6789");
console.log(p1.name);      // ✅  Output: Milan (public)
console.log(p1.getSSN());  // ✅  Output: 123-45-6789 (accessed via method)
console.log(p1.#ssn);      // ❌  Error: Private field '#ssn' must be declared in an enclosing class
```

✅ Explanation (Point-by-Point):

1. name is **public** – accessible from outside the class.
2. #ssn is **private** – cannot be accessed directly.
3. We use getSSN() as a **public method** to read the private value.

✅ Extra Tip: Simulating Protected in JS

```
class Employee {
  constructor() {
    this._salary = 50000; // Underscore means "treat as protected"
  }
}
```

🔏 Not truly protected, but a **common convention** in JS to signal: “don’t touch this outside the class or subclass.”