== vs === vs typeof

In JavaScript, ==, ===, and typeof are often used for equality comparisons and type checks.

**== (Loose Equality)**

* Performs type **coercion** before comparing values.
* If the types of the operands are different, JavaScript attempts to convert them to a common type before comparison.

console.log(5 == '5'); // true (string '5' is converted to number 5)

console.log(false == 0); // true (false is converted to 0)

console.log(null == undefined); // true (special case in JavaScript)

Problems with ==:

console.log([] == 0); // true (empty array coerced to "")

console.log('' == false); // true (both coerced to 0)

console.log(' \t\n' == 0); // true (whitespace string coerced to 0)

**=== (Strict Equality)**

* Performs **strict comparison** without type coercion.
* Both the value and the type must match for the comparison to return true.

**typeof (Type Checking)**

* Returns a string indicating the type of a value.

console.log(typeof undefined); // "undefined"

console.log(typeof null); // "object" (quirk in JavaScript)

console.log(typeof {}); // "object"

console.log(typeof []); // "object" (arrays are objects)

console.log(typeof function() {}); // "function"

Function Scope, Block Scope, and Lexical Scope

**Key Takeaways:**

1. **Function Scope** applies to var and is confined to the function in which it is declared.
2. **Block Scope** applies to let and const, restricting the variable to the block {}.
3. **Lexical Scope** ensures that a nested function retains access to variables in its outer functions, regardless of where it is invoked.

IIFE

IIFE (Immediately Invoked Function Expression):

An **IIFE** is a function that is defined and executed immediately after its declaration.

Syntax of an IIFE:

(function() { // Your code here })();

or with an arrow function:

(() => { // Your code here })();

**Why is IIFE Useful in JavaScript?**

* **Avoid polluting the global namespace**: By encapsulating variables inside an IIFE, they are not accessible globally. This prevents naming conflicts.
* **Encapsulation**: Variables inside an IIFE are private and cannot be accessed from outside, which helps in keeping the global namespace clean.
* **Execute code immediately**: You can run a function immediately, which is useful for initialization tasks (e.g., setting up variables, starting animations, etc.).