VAR	Let	Const
• `var` was traditionally used in JavaScript for variable declaration. • Variables declared with `var` are function-scoped or globally scoped. • They can be re-declared,re-assign and re-initialization can be done and updated within their scope. • **Hoisting**: Variables declared with `var` are hoisted to the top of their scope. This means you can access the variable before it's declared, but it will have an initial value of `undefined`. • **Global Scope**: If `var` is used outside of any function, it becomes a global variable, which can lead to unintended consequences and pollute the global namespace. • **Function Scope**: Variables declared with `var` are function-scoped, meaning they are visible throughout the function in which they are declared. • **No Block Scope**: Unlike `let` and `const`, variables declared with `var` are not block-scoped. They are function-scoped or globally scoped.	• Introduced in ES6 (ECMAScript 2015) to address some of the issues with `var`. • Use `let` for variables that need to be reassigned. • **Block Scope**: Variables declared with `let` are block-scoped, meaning they are limited to the block (`{}`) in which they are defined. • **No Hoisting Issues**: Unlike `var`, variables declared with `let` are not hoisted to the top of their block. They behave as you would expect them to. • **Can Be Updated**: Variables declared with `let` can be reassigned a new value within their scope. • **Cannot Be Redefined**: You cannot declare another variable with the same name in the same scope using `let`.	• Also introduced in ES6, `const` is used to declare constants, which are read-only. • `const` for variables that do not need reassignment. • **Constant Value**: Variables declared with `const` must be initialized with a value, and that value cannot be changed through reassignment. • **Block Scope**: Like `let`, variables declared with `const` are block-scoped. • **Immutable Binding**: Although the value assigned to a `const` variable is immutable, it does not mean the value itself is immutable if it's an object or array. You can change the properties or elements of the object or array. • **Must Be Initialized**: You must assign a value when declaring a `const` variable. Unlike `let`, you cannot declare a `const` variable without initializing it.