Variables (var, let, const) - differences and scope

In JavaScript, variables can be declared using `var`, `let`, or `const`. Each has different behaviors regarding **scope**, **hoisting**, and **reassignment**. Understanding these differences is critical for writing clean and bug-free code.

🔷 1. var

✅ Characteristics:

- **Function-scoped**: Accessible throughout the function in which it is declared.
- **Hoisted**: Declarations are moved to the top of their scope and initialized with `undefined`.
- **Re-declarable**: Can be re-declared in the same scope without an error.
- **Reassignable**: You can reassign values.

📦 Example:

```
function demoVar() {
  console.log(a); // undefined (due to hoisting)
  var a = 10;
  console.log(a); // 10
}
demoVar();
```

❌ Drawback:

`var` can lead to **unexpected bugs** due to hoisting and lack of block scope.

🔷 2. let

✅ Characteristics:

- **Block-scoped**: Accessible only within the `{}` block where it is declared.
- **Hoisted**: But **not initialized** — accessing before declaration causes a **ReferenceError** (known as the *Temporal Dead Zone*).
- **Not re-declarable**: Cannot be re-declared in the same scope.
- **Reassignable**: You can reassign values.

📦 Example:

```
function demoLet() {
  // console.log(b); // ReferenceError
  let b = 20;
  console.log(b); // 20
}
demoLet();
```

🔷 3. const

✅ Characteristics:

- **Block-scoped**: Like `let`.
- **Hoisted**: Like `let`, but also in the *Temporal Dead Zone*.
- **Not re-declarable**.
- **Not reassignable**: Once assigned, cannot change the binding.

📦 Example:

```
const c = 30;
// c = 40; // TypeError: Assignment to constant variable.
```

> 🔷 Note: const only makes the variable binding immutable, not the value itself if it's an object.

```
const obj = { name: "Abhi" };
obj.name = "Gurjar"; // Allowed
console.log(obj.name); // Gurjar
```

📋 Summary Table

Feature	`var`	`let`	`const`
Scope	Function	Block	Block
Hoisting	Yes (undefined)	Yes (TDZ)	Yes (TDZ)
Re-declarable	Yes	No	No
Reassignable	Yes	Yes	No
Best Use Case	Legacy code	General use	Constants

✅ When to Use What?

- Prefer `*const**` by default.
- Use `*let**` only if you need to reassign the variable later.
- Avoid `*var**` in modern JavaScript — it's mostly outdated.

Let me know if you want visual illustrations or comparison examples for real use cases.