In JavaScript, `let`, `var`, and `const` are used to declare variables. These keywords have different scopes, behaviors, and use cases. Here are the main differences between them:

1. Scope:

- `var`: has function-level scope, meaning that a variable declared with `var` inside a function is only accessible within that function.

- `let` and `const`: have block-level scope, meaning that a variable declared with `let` or `const` inside a block (e.g., within curly braces {}) is only accessible within that block.

2. Hoisting:

- `var`: is hoisted to the top of the function or global scope, meaning that a variable declared with `var` can be used before it is declared in the code.

- `let` and `const`: are not hoisted, meaning that a variable declared with `let` or `const` cannot be used before it is declared in the code.

3. Reassignment:

- `var` and `let`: allow reassignment, meaning that the value of a variable declared with `var` or `let` can be changed.

- `const`: does not allow reassignment, meaning that the value of a variable declared with `const` cannot be changed.

4. Initialization:

- `var` and `let`: can be declared without being initialized. If a variable is not initialized, its value will be undefined.

- `const`: must be initialized at the time of declaration. If a variable declared with `const` is not initialized, an error will occur.