let and const

Download Demo Code <../js-let-const-demo.zip>

reviewing *var*

var

- We use the ***var*** keyword to declare variables
- When defined in a function, the ***var*** keyword scopes a variable to that function
- ***var*** will hoist to the top of the scope it is defined in
- You can redeclare and reassign values with ***var***

```
var instructor = "Colt"; // accessible everywhere!

function greet(){
  var message = "Hello!"; // scoped to the greet function
}
```

let

The ***let*** keyword creates a **block-scoped** variable: a variable that only exists inside a code block.

What Is A Code Block?

Essentially any pair of curly braces (outside of object syntax).

```
{
  // this is a code block

  let x = 5;

  var y = 10;

}

console.log(x);
// ReferenceError: x is not defined

console.log(y);
// 10
```

Where Are Code Blocks Commonly Used?

You’ll mostly use code blocks in ***for*** loops and ***if*** statements.

```
if (x > 10) {
  let happy = true; // happy lives ONLY in this code block
}
// can't use it outside the block
console.log(happy); // ReferenceError: happy is not defined
```

An Example

```
for (var i = 1; i < 4; i++) {
  console.log(i);
}

// 1
// 2
// 3

console.log(i);
// 4
```

```
for (let i = 1; i < 4; i++) {
  console.log(i);
}

// 1
// 2
// 3

console.log(i); // ReferenceError: i is not defined
```

More About *let*

It can be reassigned but not redeclared (unlike ***var***).

```
let z = 5;
z = 25;
let z = 10;
// SyntaxError: Identifier 'z' has already been declared
```

const

The ***const*** keyword prevents a variable from ever being reassigned or redeclared.

```
const PI = 3.14;

PI = 15; // TypeError: Assignment to constant variable

const PI = 5;  // SyntaxError
```

const is also block-scoped, like ***let***.

```
{
  const x = 10;
}

console.log(x); // ReferenceError: x is not defined
```

Comparison of Variable Declaration Keywords

Keyword	Can Reassign	Can Redeclare	Can Mutate	Scope Rules
var	yes	yes	yes	function scope
let	yes	no	yes	block scope
const	no	no	yes	block scope

What about *var*?

- There’s really no need to use it
- Just be careful of block scoping with ***let***