

[Tutorials ▼](#)[References ▼](#)[Exercises ▼](#)[Menu ▼](#)[Website NEW](#)[Paid Courses](#)[Log in](#)[HTML](#)[CSS](#)[JAVASCRIPT](#)[SQL](#)[PYTHON](#)[Data](#)

# JavaScript Let

[< Previous](#)[Next >](#)

The **let** keyword was introduced in [ES6 \(2015\)](#).

Variables defined with **let** cannot be Redeclared.

Variables defined with **let** must be Declared before use.

Variables defined with **let** have Block Scope.

## Cannot be Redeclared

Variables defined with **let** cannot be **redeclared**.

You cannot accidentally redeclare a variable.

With **let** you can not do this:

## Example

```
let x = "John Doe";

let x = 0;

// SyntaxError: 'x' has already been declared
```

With **var** you can:

## Example

```
var x = "John Doe";

var x = 0;
```

---

# Block Scope

Before ES6 (2015), JavaScript had only **Global Scope** and **Function Scope**.

ES6 introduced two important new JavaScript keywords: **let** and **const**.

These two keywords provide **Block Scope** in JavaScript.

Variables declared inside a { } block cannot be accessed from outside the block:

## Example

```
{
  let x = 2;
```

```
}  
// x can NOT be used here
```

Variables declared with the **var** keyword can NOT have block scope.

Variables declared inside a { } block can be accessed from outside the block.

## Example

```
{  
  var x = 2;  
}  
// x CAN be used here
```

---

## Redeclaring Variables

Redeclaring a variable using the **var** keyword can impose problems.

Redeclaring a variable inside a block will also redeclare the variable outside the block:

## Example

```
var x = 10;  
// Here x is 10  
  
{  
  var x = 2;  
  // Here x is 2  
}  
  
// Here x is 2
```

### Try it Yourself »

Redeclaring a variable using the **let** keyword can solve this problem.

Redeclaring a variable inside a block will not redeclare the variable outside the block:

## Example

```
let x = 10;  
// Here x is 10  
  
{  
  let x = 2;  
  // Here x is 2  
}  
  
// Here x is 10
```

### Try it Yourself »

---

## Browser Support

The **let** keyword is not fully supported in Internet Explorer 11 or earlier.

The following table defines the first browser versions with full support for the **let** keyword:

Chrome 49	Edge 12	Firefox 44	Safari 11	Opera 36
Mar, 2016	Jul, 2015	Jan, 2015	Sep, 2017	Mar, 2016

# Redeclaring

Redeclaring a JavaScript variable with **var** is allowed anywhere in a program:

## Example

```
var x = 2;  
// Now x is 2  
  
var x = 3;  
// Now x is 3
```

**Try it Yourself »**

With **let**, redeclaring a variable in the same block is NOT allowed:

## Example

```
var x = 2;    // Allowed  
let x = 3;    // Not allowed  
  
{  
  let x = 2;  // Allowed  
  let x = 3   // Not allowed  
}  
  
{  
  let x = 2;  // Allowed  
  var x = 3   // Not allowed  
}
```

Redeclaring a variable with **let** , in another block, IS allowed:

## Example

```
let x = 2;    // Allowed

{
  let x = 3;  // Allowed
}

{
  let x = 4;  // Allowed
}
```

[Try it Yourself »](#)

---

## Let Hoisting

Variables defined with **var** are **hoisted** to the top and can be initialized at any time.

Meaning: You can use the variable before it is declared:

## Example

This is OK:

```
carName = "Volvo";
var carName;
```

[Try it Yourself »](#)

If you want to learn more about hoisting, study the chapter [JavaScript](#)

## Hoisting.

Variables defined with **let** are also hoisted to the top of the block, but not initialized.

Meaning: Using a **let** variable before it is declared will result in a **ReferenceError** :

## Example

```
carName = "Saab";  
let carName = "Volvo";
```

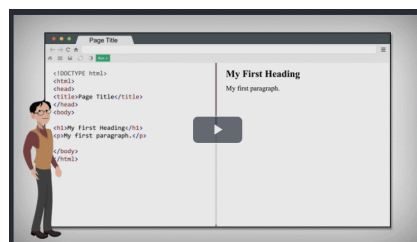
Try it Yourself »

◀ Previous

Next ▶

NEW

We just launched  
W3Schools videos



Explore now