

# JS\_Basics-S1-Variables

variables

Training Clarusway

Pear Deck - August 6, 2022 at 8:57AM

## Part 1 - Summary

Use this space to summarize your thoughts on the lesson

## Part 2 - Responses

Slide 1



JavaScript  
Variables  
Session-1



CLARUSWAY  
HELPING YOU GROW YOUR BUSINESS

Use this space to take notes:

## Slide 2

### Table of Contents

- ▶ JavaScript Variables
- ▶ The Assignment Operator
- ▶ Naming Rules
- ▶ Reserved Words in JavaScript
- ▶ let vs var vs const
- ▶ Stack and Heap

CLARUSWAY  
WAY TO REFINEMENT YOURSELF

Use this space to take notes:

## Slide 3

Play  
**Kahoot!**

CLARUSWAY  
WAY TO REFINEMENT YOURSELF

Link(s) on this slide:

- <https://create.kahoot.it/details/2-variables/f108964e-1be0-4f3a-a9ae-9ded0a17434f>

Use this space to take notes:

## Slide 4



### 1 What is Variable?



CLARUSWAY  
WAY TO REBIRTH YOURSELF

Use this space to take notes:

## Slide 5

## Your Response

### ► What is Variables?



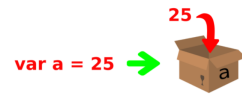
Students, write your response!

Press Deck Interactive Slide

Use this space to take notes:

## Slide 6

### ► What is Variable?



var a = 25;

Variables are used to store for data values

CLARUSWAY  
WAY TO REFINEMENT YOURSELF



Use this space to take notes:

## Slide 7

### ► What is Variable?



```
var myNumber = 3;
```

```
var quantity;
```

VARIABLE KEYWORD      VARIABLE NAME

In this example,  
\* var is a variable keyword.  
\* myNumber is a variable.  
\* 3 is a value.

```
quantity = 3;
```

ASSIGNMENT OPERATOR

VARIABLE NAME      VARIABLE VALUE

Warning ! : JavaScript is case sensitive. This means that the variables myNumber, mynumber or MYNUMBER are not same variables. All of them are different variables.

CLARUSWAY  
WAY TO REFINEMENT YOURSELF



Use this space to take notes:



## 2 The Assignment Operator

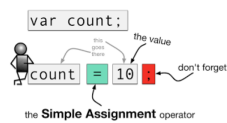
CLARUSWAY  
WAY TO REFINEMENT YOURSELF

Use this space to take notes:

## ► The Assignment Operator



- In JavaScript, the equal sign (=) is used for the assignment operator
- We can store a value in a variable with the assignment operator.



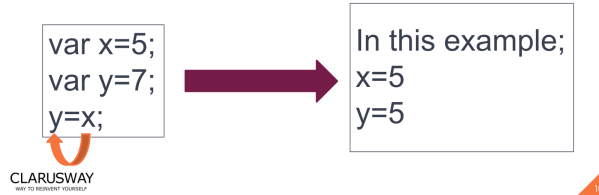
CLARUSWAY  
WAY TO REFINEMENT YOURSELF



Use this space to take notes:

## ► The Assignment Operator »

Assignment always goes from right to left



Use this space to take notes:



## 3 ► Naming Rules



CLARUSWAY  
WAY TO REFINEMENT YOURSELF

Use this space to take notes:

## Slide 12

### ► Naming Rules

- Names can be composed of letters, digits, underscores, and dollar signs
- Numbers are not allowed as the first character
- The first character must be:  
a letter, an underscore (\_), a dollar sign (\$)
- JavaScript names must not contain spaces, mathematical or logical operators
- Reserved words cannot be used as names

CLARUSWAY  
WAY TO REFINEMENT YOURSELF

4thofCats	❌	variable	✅
Your name	❌	result	❌
their surname	❌	sh@rt@w	❌
first_color	✅	\$!variable	✅

12

Use this space to take notes:

## Slide 13

### ► Naming Rules

Reserved words cannot be used as names

JavaScript Reserved Words				
abstract	Arguments	await	boolean	break
byte	Case	catch	char	class
const	continue	debugger	default	delete
do	double	else	enum	eval
export	extends	false	final	finally
float	for	function	goto	if
implements	import	in	instanceof	int
interface	let	long	native	new
null	package	private	protected	public
return	short	static	super	switch
synchronized	this	throw	throws	transient
true	try	typeof	var	void
volatile	while	with	yield	

CLARUSWAY  
WAY TO REFINEMENT YOURSELF

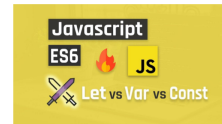
13

Use this space to take notes:



## 4 *let vs var vs const*

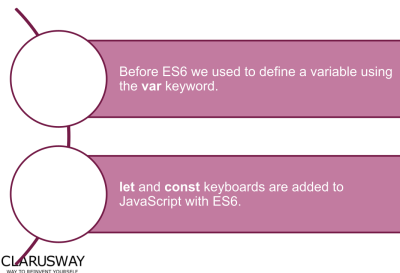
CLARUSWAY  
WAY TO REFINEMENT YOURSELF



Use this space to take notes:



## ▶ *let vs var vs const*



CLARUSWAY  
WAY TO REFINEMENT YOURSELF



Use this space to take notes:



## Slide 16

### ► Scope



- Three types of scope:
  - ◆ Global scope
  - ◆ Function scope
  - ◆ Block scope
- Global scope
  - Outside any function
  - Variables can be accessed from **anywhere in the program**



Use this space to take notes:

## Slide 17

### ► Scope Function Scope



- Variables defined anywhere *inside* a function are **local** to that function
- Can be used anywhere inside that function
- Cannot be used outside that function

```
// code here can NOT use myNumber

function myFunction() {
  var myNumber = 42;
  //code here CAN use myNumber
}

// code here can NOT use myNumber
```



Use this space to take notes:

## Slide 18

### ► Scope Block Scope

- To limit a variable to its block inside the function, use *let*

```
function fn(num){  
  if (num > 5){  
    var newNum = 5;  
  }  
  // newNum CAN be accessed here  
}
```

```
function fn(num){  
  if (num > 5){  
    let newNum = 5;  
  }  
  // newNum can NOT be accessed here  
}
```

CLARUSWAY  
WAY TO REFINEMENT YOURSELF



Use this space to take notes:

## Slide 19

### ► let vs var

- At global and function scopes, *let* and *var* work *almost* the same
- *var* supports redeclaration, while *let* does not
- Both support re-assignment. Use *const* to disallow it
- *let* is more like regular variables in other languages  
Preferred over *var*

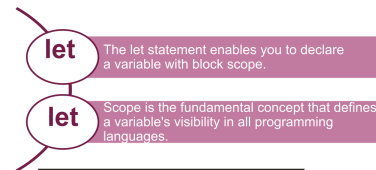
CLARUSWAY  
WAY TO REFINEMENT YOURSELF



Use this space to take notes:

## Slide 20

### ► let vs var vs const



```
<script>
var a = 10;
{
  let b = 3;
}
console.log("a = " + a);
console.log("b = " + b); // generates an error
//script.js
```

CLARUSWAY  
WAY TO REFINEMENT YOURSELF



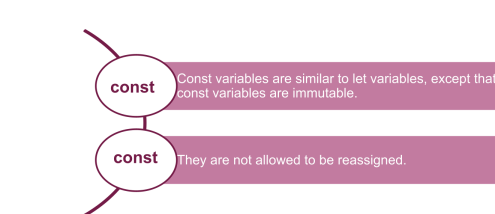
```
Uncaught ReferenceError: b is not defined
at first.html:11
```

20

Use this space to take notes:

## Slide 21

### ► let vs var vs const



CLARUSWAY  
WAY TO REFINEMENT YOURSELF

21

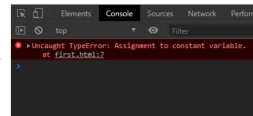
Use this space to take notes:

## Slide 22

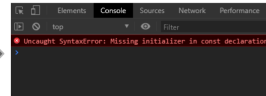
### ► let and const



```
<script>  
const x = 5;  
x = 7; // generates an error  
</script>
```



```
<script>  
const x; // generates an error  
x = 7;  
</script>
```



CLARUSWAY  
WAY TO REBIRTH YOURSELF



Use this space to take notes:

## Slide 23



### 4 Stack and Heap

CLARUSWAY  
WAY TO REBIRTH YOURSELF

Use this space to take notes:

## Slide 24

### ► Stack and Heap

Variables in JavaScript (and most other programming languages) are stored in two places: stack and heap.

- A stack is usually a continuous region of memory allocating local context for each executing function.
- Heap is a much larger region storing everything allocated dynamically.
- This separation is useful  
Stack is more protected and faster, no need for dynamic garbage collection.

CLARUSWAY  
WAY TO REFINEMENT YOURSELF



Use this space to take notes:

## Slide 25

★ Primitive values example:

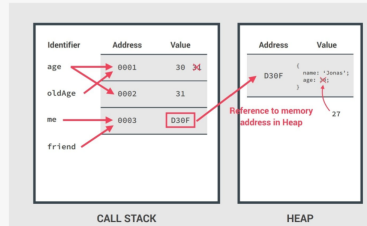
```
let age = 30;
let oldAge = age;
age = 31;
console.log(age); // 31
console.log(oldAge); // 30
```

★ Reference values example:

```
const me = {
  name: 'Jonas',
  age: 38
};
const friend = me;
friend.age = 27;

console.log('Friend:', friend);
// { name: 'Jonas', age: 27 }

console.log('Me:', me);
// { name: 'Jonas', age: 27 }
```



CLARUSWAY  
WAY TO REFINEMENT YOURSELF

Source: Jonas Schmedtmann, The Complete JavaScript Course 2020: From Zero to Expert!



Use this space to take notes:

**THANKS!**

**Any questions?**



CLARUSWAY  
WAY TO RESIDENT YOURSELF



Use this space to take notes: