# JavaScript Data Types

## Session-2

CLARUSWAY
WAY TO REINVENT YOURSELF

# Table of Contents

- Data Types
- The typeof Operator
- Undefined & Null
- Numbers
- Strings
- Booleans

```
<script>
if(age > 19){
    alert("Adult");
} else{
    alert("Teenager");
}
</script>
```

# Did you finish Javascript Core pre-class material?

Students choose an option

Pear Deck Interactive Slide
Do not remove this bar

# Play Kahoot!

1

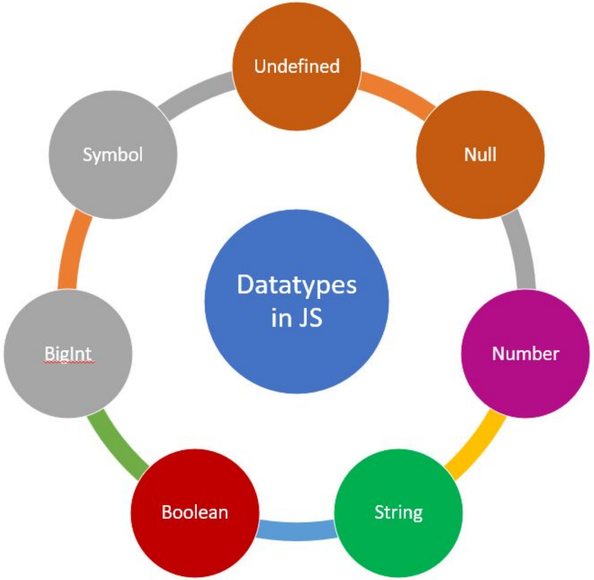
# Data Types

# Data Types

JavaScript variables can hold numerous data types, such as numbers, strings, booleans, and more

It is required to declare a variable for creating

In JavaScript, we can declare a variable with the var, let or const keyword



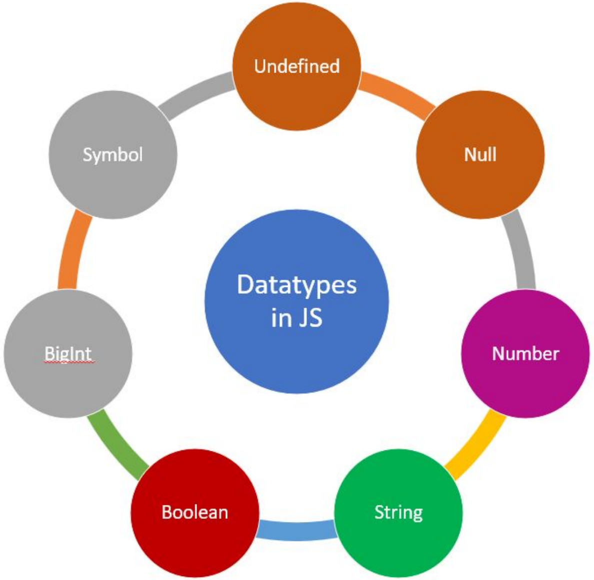
# Data Types

let myNumber;

In the example above, myNumber value has no value. (actually, the value of myNumber is undefined)

let myNumber  = 10;

It is necessary to use the equal sign for assigning a value to the variable

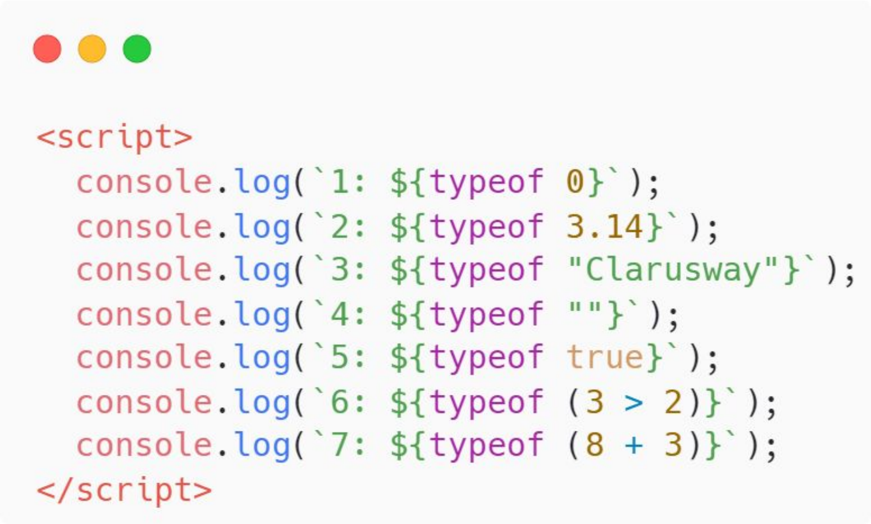


2

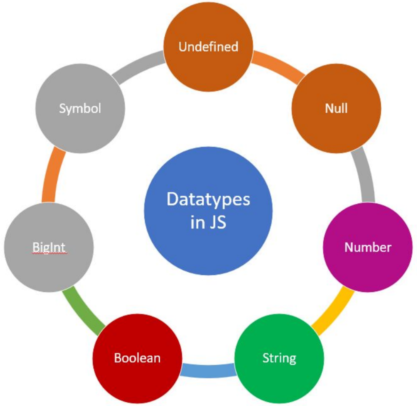
# The `typeof` Operator

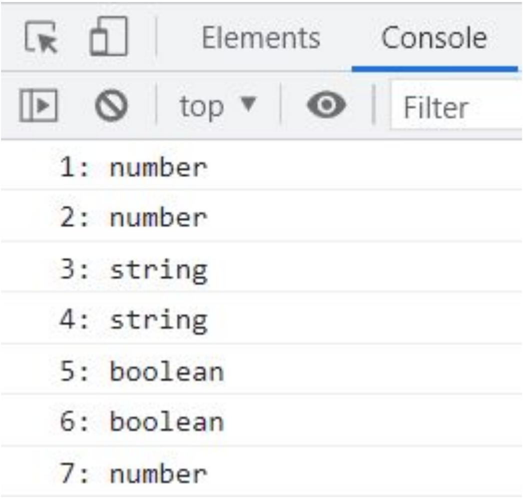
# The *typeof* Operator

To find the type of a JavaScript variable, we can use the JavaScript typeof operator



```
<script>
  console.log(`1: ${typeof 0}`);
  console.log(`2: ${typeof 3.14}`);
  console.log(`3: ${typeof "Clarusway"}`);
  console.log(`4: ${typeof ""}`);
  console.log(`5: ${typeof true}`);
  console.log(`6: ${typeof (3 > 2)}`);
  console.log(`7: ${typeof (8 + 3)}`);
</script>
```





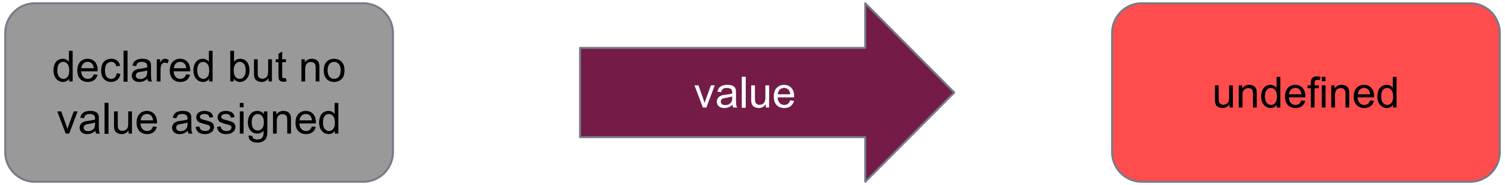
```
1: number
2: number
3: string
4: string
5: boolean
6: boolean
7: number
```

CLARUSWAY
WAY TO REINVENT YOURSELF

3

# undefined

# undefined

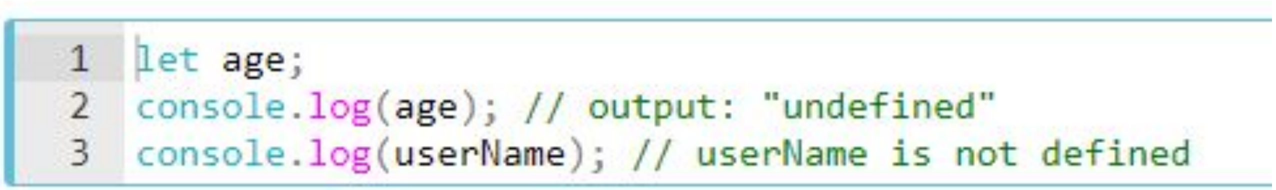


declared but no value assigned

value

undefined

undefined is the initial state of a variable if it has no assigned value.



```
let age;
console.log(age); // output: "undefined"
console.log(userName); // userName is not defined
```

4

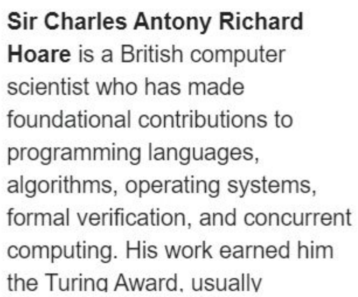
# null

NULL=🤔

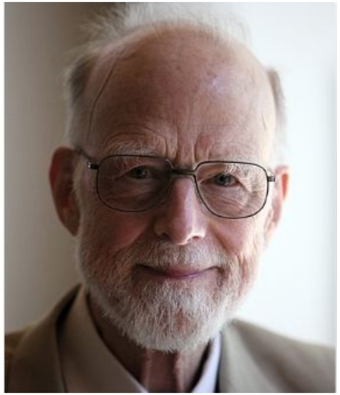
# null

- ***null*** is a special type among other data types.
- ***null*** is a special keyword to indicate intentional absence of value,
- ***null*** is a primitive type.
- Garbage collection mechanism uses ***null*** to release unused memory.

# null



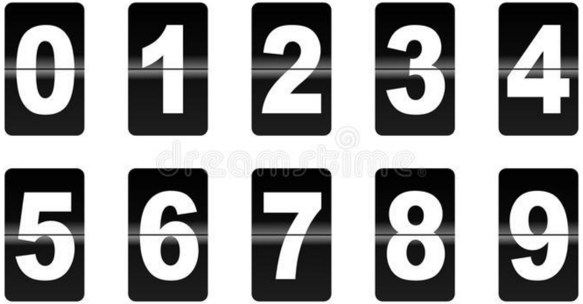
**Sir Charles Antony Richard Hoare** is a British computer scientist who has made foundational contributions to programming languages, algorithms, operating systems, formal verification, and concurrent computing. His work earned him the Turing Award, usually



in 1995, Tony Hooare stated:

I call it my ***billion-dollar mistake***. It was the invention of the null reference in 1965. At that time, I was designing the first comprehensive type system for references in an object oriented language (ALGOL W). My goal was to ensure that all use of references should be absolutely safe, ….. *This has led to innumerable errors, vulnerabilities, and system crashes, which have probably caused a billion dollars of pain and damage in the last forty years.*

# 5 Numbers



# Numbers

Unlike many other programming languages, JavaScript has only one type of number

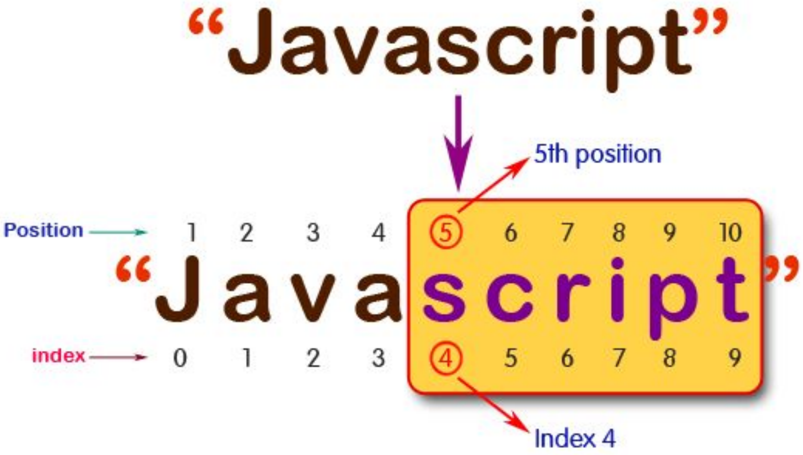
```
let x = 7 ;
```

Numbers can be written with decimals

```
let y = 33.33 ;
```

6

# Strings


"Javascript"
5th position
Position
1 2 3 4 5 6 7 8 9 10
"Javascript"
index
0 1 2 3 4 5 6 7 8 9
Index 4

# Strings

- Strings defined as the sequence of characters
- In JavaScript, strings are used for storing and manipulating text
- A string can be any content that shows up inside quotes. We can use single or double-quote
- We can use quotes inside a string unless they don't match the quotes encompassing the string
- If we need to use quotes inside the string, we can use backslash ( \ ) escape character

```
let myName = "John";

let myAge = "36";

let company = `Clarusway`;

let text2 = 'She said, "Go ahead"';

let text = "He said, \"I am a new
programmer.\"";
```

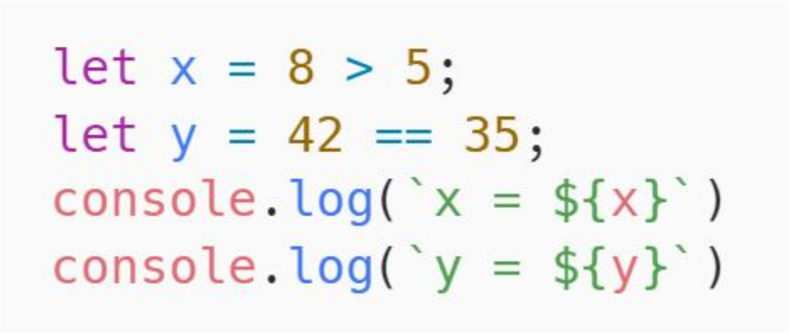
7

# Booleans

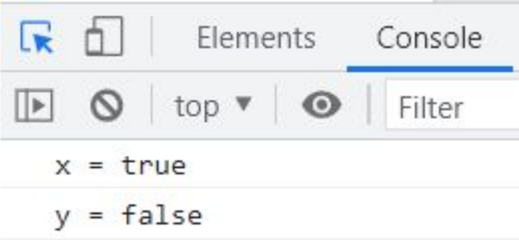
# Booleans

A JavaScript Boolean can have one of two values, either true or false

Boolean data type is useful when we need to have one of two values, such as true/false, yes/no, etc.



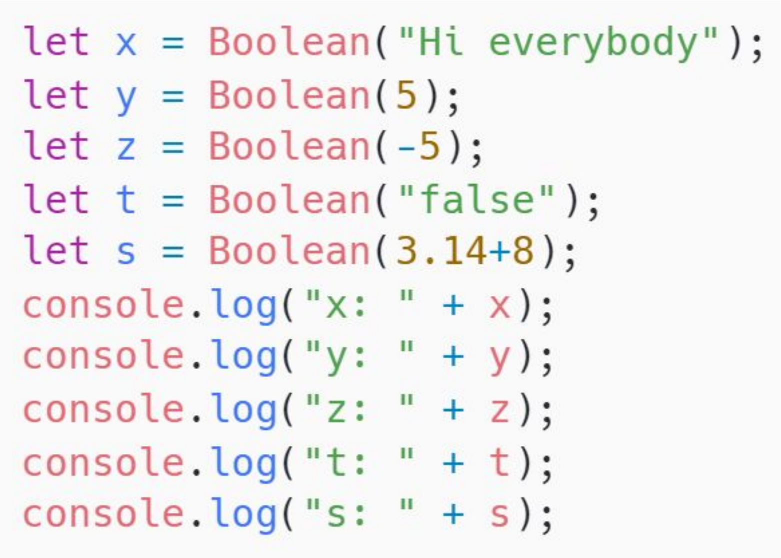
```
let x = 8 > 5;
let y = 42 == 35;
console.log(`x = ${x}`)
console.log(`y = ${y}`)
```



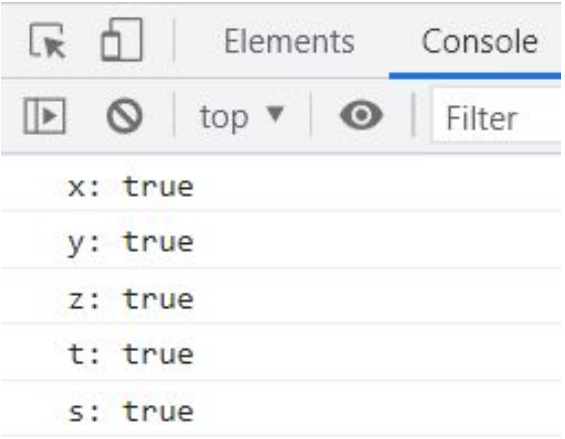
```
x = true
y = false
```

# Booleans

The Boolean value of everything with a "value" is true.



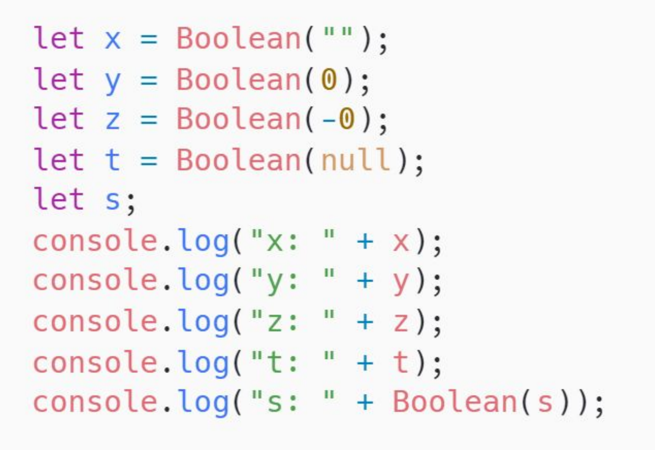
```
let x = Boolean("Hi everybody");
let y = Boolean(5);
let z = Boolean(-5);
let t = Boolean("false");
let s = Boolean(3.14+8);
console.log("x: " + x);
console.log("y: " + y);
console.log("z: " + z);
console.log("t: " + t);
console.log("s: " + s);
```



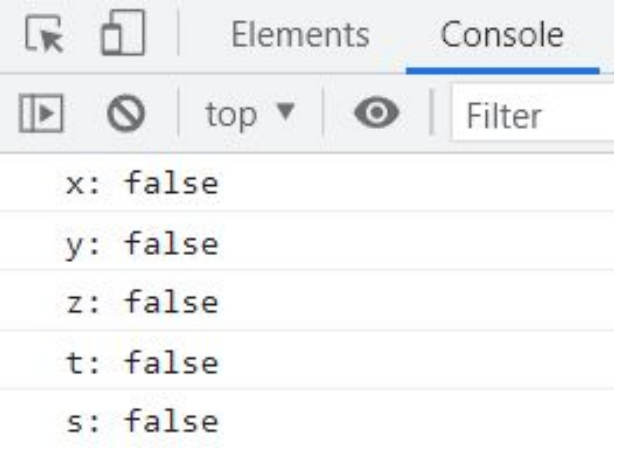
```
x: true
y: true
z: true
t: true
s: true
```

# Booleans

- The Boolean value of everything without a "value" is false
- The Boolean value of 0, null, empty string, undefined, is false



```
let x = Boolean("");
let y = Boolean(0);
let z = Boolean(-0);
let t = Boolean(null);
let s;
console.log("x: " + x);
console.log("y: " + y);
console.log("z: " + z);
console.log("t: " + t);
console.log("s: " + Boolean(s));
```



```
x: false
y: false
z: false
t: false
s: false
```

# 8 Objects

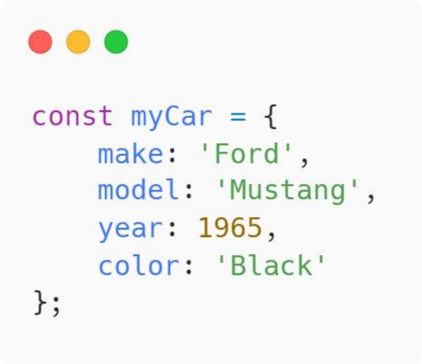
{ }

# Objects

- In JavaScript, an ***object*** is a standalone entity, with properties and type.

- Compare it with a car, for example. A car is an object, with properties.
  - A car has a make, a model, year, a color, etc.

- In the same way, JavaScript ***objects*** can have properties, which define their characteristics.



```
const myCar = {
    make: 'Ford',
    model: 'Mustang',
    year: 1965,
    color: 'Black'
};
```

# THANKS!

## Any questions?