

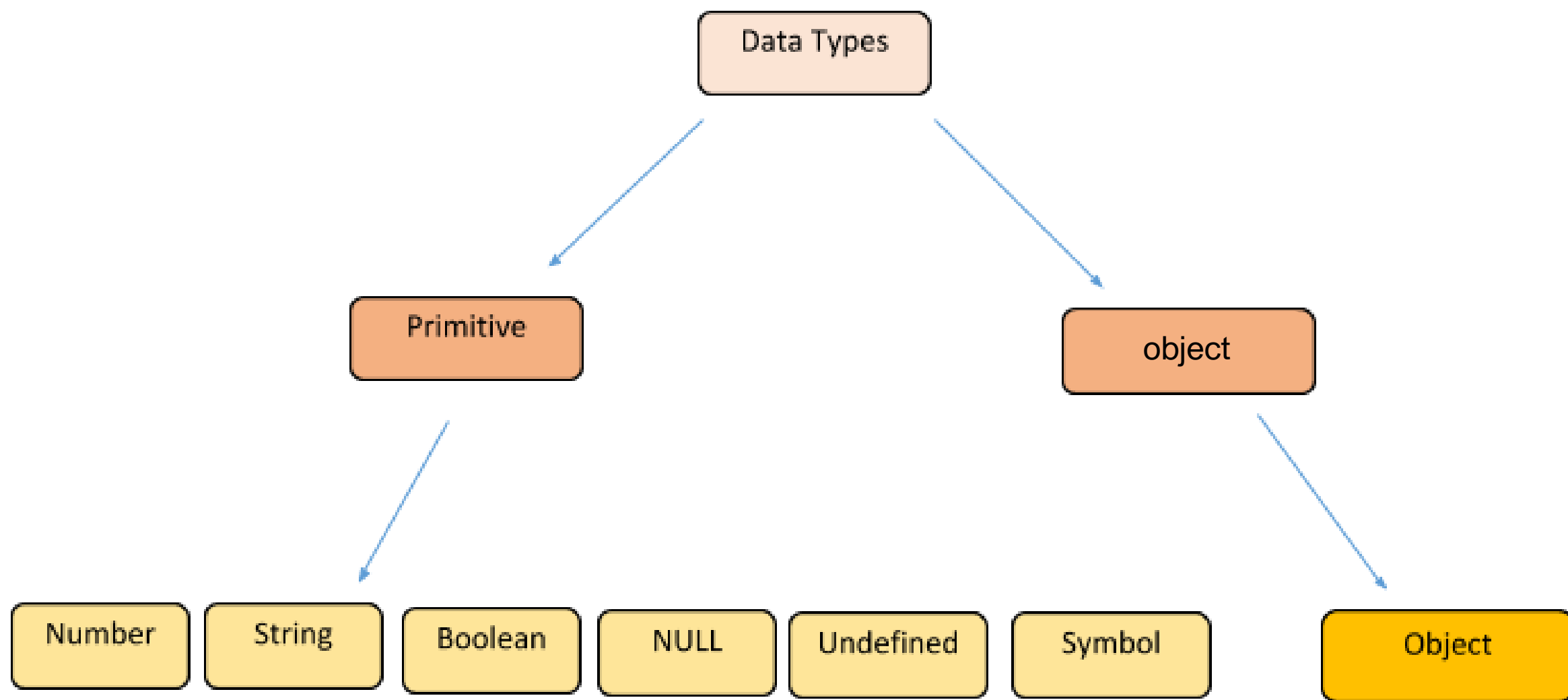


You don't know JS Yet  
Mina

# Value forms

1. Primitive values

2. Object values



# Value forms

1. Primitive values
2. Object values

literal

variable



# Examples of literals in JavaScript

- String literal: "Hello, World!"
- Number literal: 42
- Boolean literal: true or false
- Array literal: [1, 2, 3, 4]
- Object literal: { name: "Mina", age: 33 }

# Three ways to define a string:

Single quotes (')

Double quotes (")

Backticks (`)



# Single quotes (')

```
console.log ('My name is ${ firstName }.');
```



```
My name is ${ firstName }.
```

# Double quotes ("")

```
console.log ("My name is ${ firstName }.");
```



```
My name is ${ firstName }.
```



# Backticks(` `)

```
console.log (`My name is ${ firstName }.`);
```



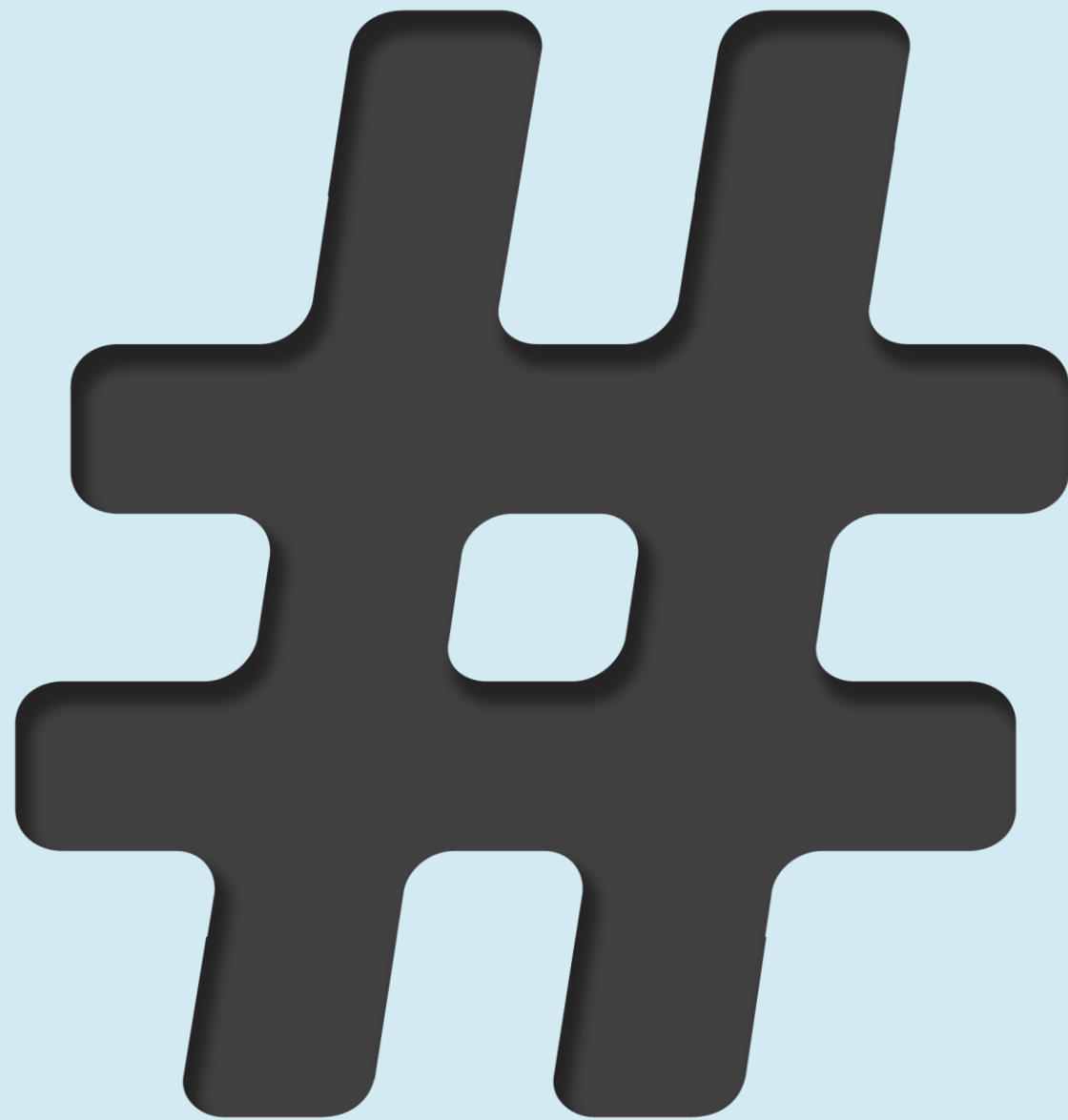
My name is Mina.

# Backticks(`)`)

```
console.log (`My name is Mina .`);
```



My name is Mina.



**numbers.js**

# BigInt

JAVASCRIPT:  
BIGINT

[illegible]

9007199254740991n

9007199254740991n

9007199254740991n

9007199254740991n

✖ ▶ Uncaught TypeError: Cannot mix BigInt and other types, use explicit conversions  
at script.js:169:7





Non-zero value



null



0



undefined



*What is it?*

*Unique?*



*Where is it used?*

*Properties?*

**Symbol, what the heck?**

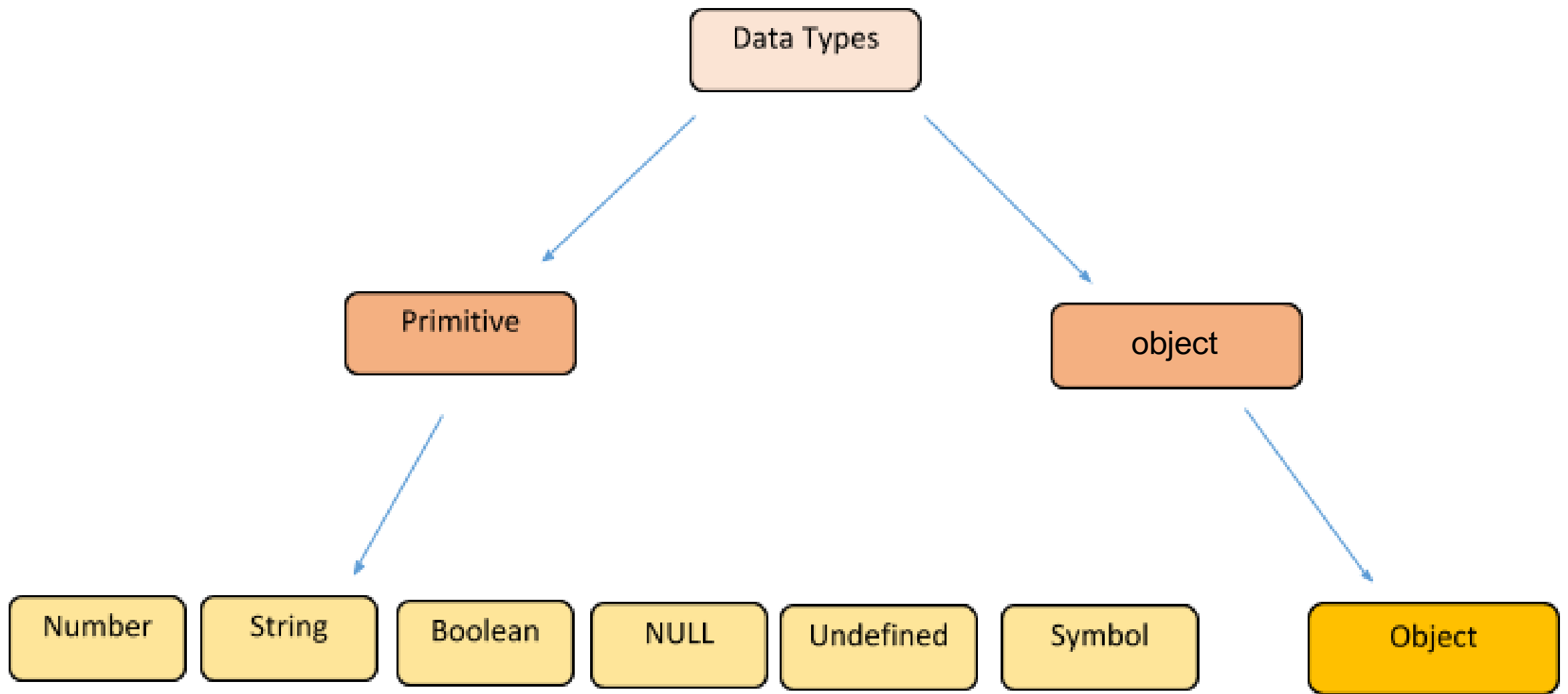
In JavaScript, a symbol is a primitive data type and is used to create unique identifiers. Symbols are immutable (cannot be changed) and are guaranteed to be unique.

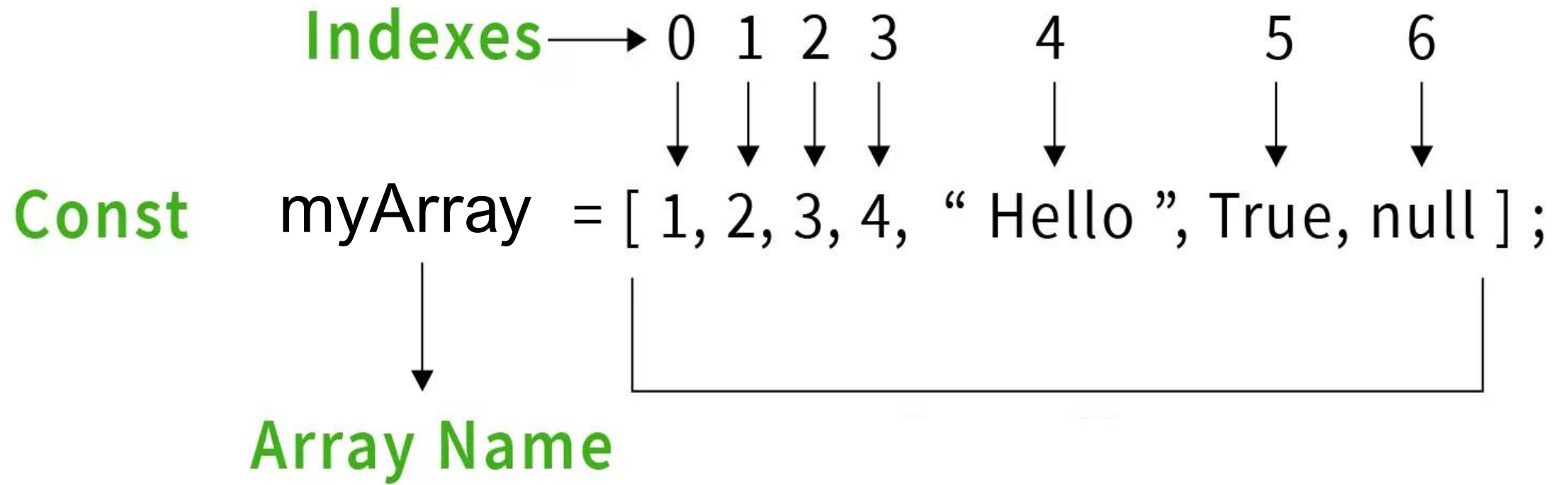
```
43  const mySecondSymbol = Symbol("abc");
44  console.log(mySecondSymbol);
45
46  const myThirdSymbol = Symbol("abc");
47
48  if (mySecondSymbol === myThirdSymbol) {
49    console.log(true);
50  } else {
51    console.log(false);
52  }
```

Symbol(abc)

false







```
let myCup = {  
  color: "transparent",  
  volume: 1,  
  weight: 0.5  
}
```

# **Variables**

# A Simple Explanation of JavaScript Variables: **const, let, var**



**const**



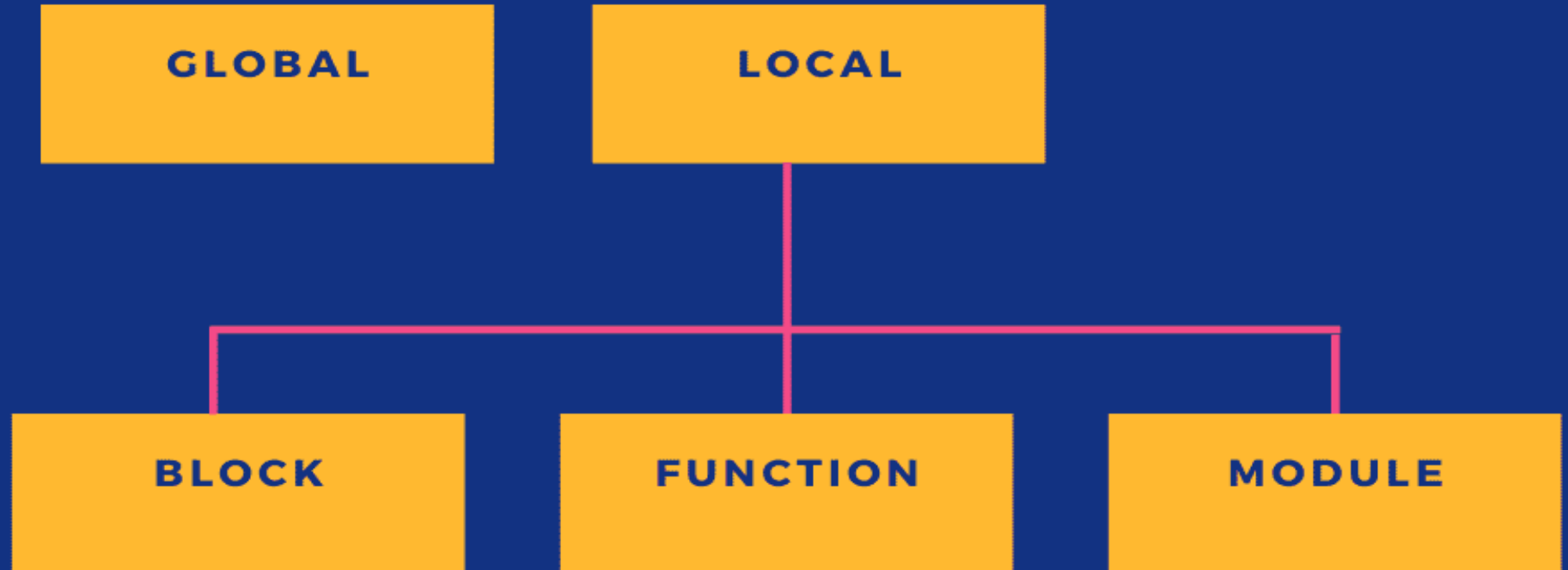
**let**

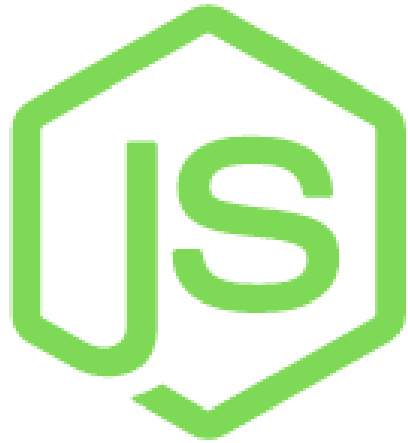


**var**



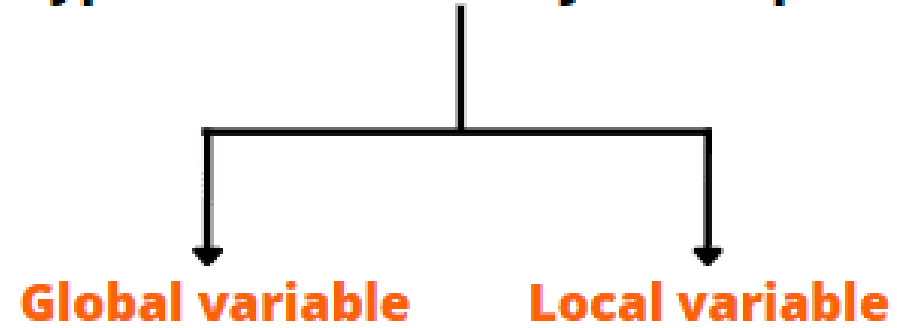
# Scope



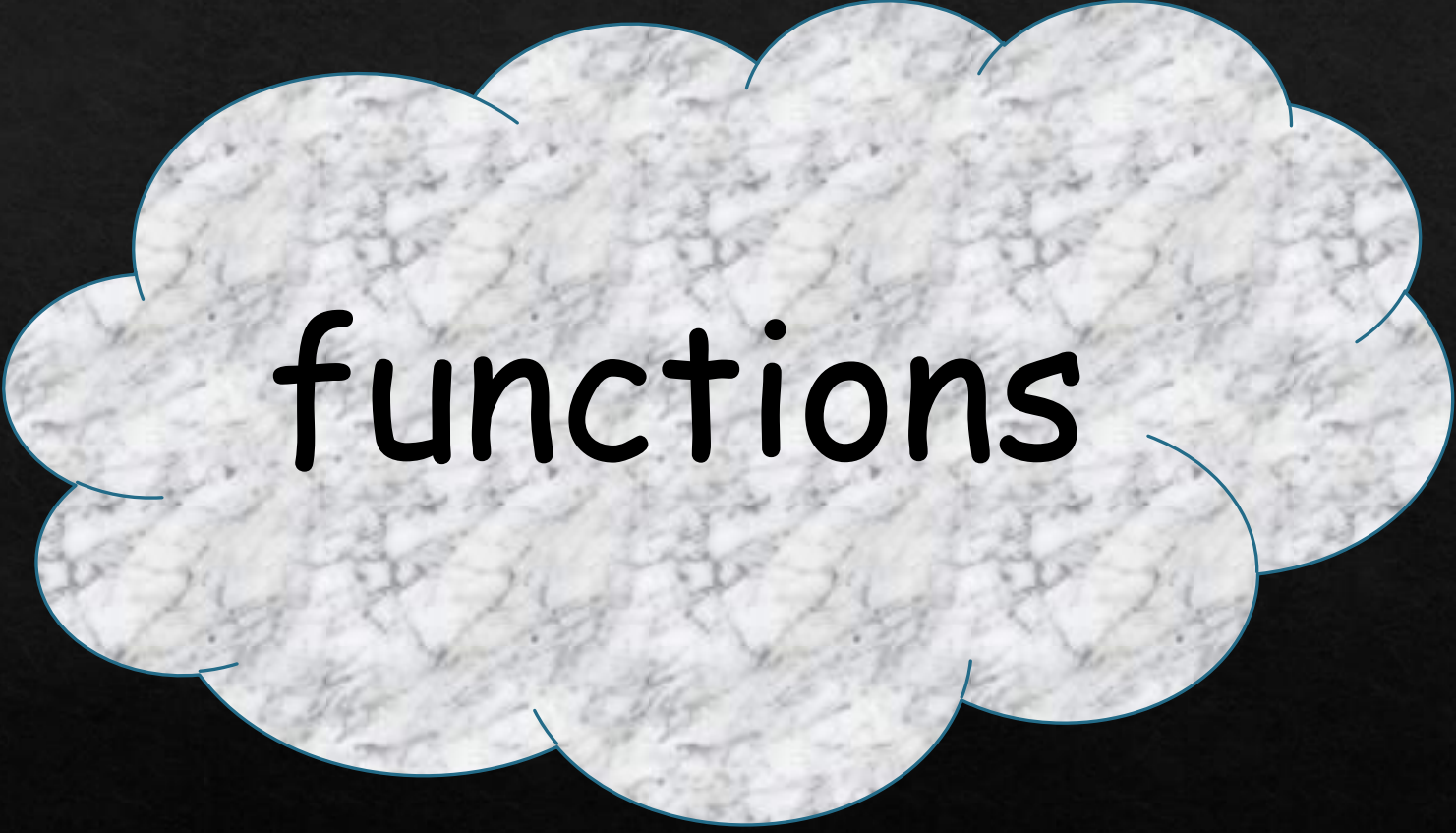


# Variables

## Types of variables in JavaScript



Sciencetech Easy



functions



Statements

در یک زبان به مجموعه ای از کلمات، اعداد و  
عملگرها که یک کار خاص را انجام می‌دهند،

Statement

میگویند



```
let x, y, z;           // Statement 1
x = 5;                 // Statement 2
y = 6;                 // Statement 3
z = x + y;             // Statement 4
```

break  
continue  
for  
for...in  
function  
if...else  
new  
return  
var  
while  
with



Expressions

در جاوا اسکریپت  
یک

Statement

از یک یا چند

Expression

ساخته شده است.

Expression

می تواند

یک متغیر،

یک

value

یا مجموعه ای از

ها value

باشد که با اپراتورها ترکیب شده اند.

For example:

```
a = b * 2;
```

This statement has four expressions in it:

- 2 is a *literal value expression*.
- b is a *variable expression*, which means to retrieve its current value.
- b \* 2 is an *arithmetic expression*, which means to do the multiplication.
- a = b \* 2 is an *assignment expression*, which means to assign the result of the b \* 2 expression to the variable a (more on assignments later).

A general expression that stands alone is also called an *expression statement*, such as the following:

```
b * 2;
```



A light brown, corkboard-textured cloud shape with a thin blue outline, centered on a dark gray background. The word "functions" is written in a black, lowercase, sans-serif font in the center of the cloud.

functions



a set of statements that performs a task or  
calculates a value

