JavaScript

The Game

Ahmed Moawad

Sr. Frontend Engineer @ **tajawal**

/in/ahmedmoawad

/AhmedMoawad

About Me

Ahmed Moawad

Tajawal
Sr. Frontend Engineer

ITI
Teaching Assistant

Sooft
Fullstack JavaScript Engineer

JS

Course Prerequisites

JS

Course Objectives

Learn about JavaScript,
its uses and really understand it.

Learn how to build dynamic and
interactive websites.

Make you fall in love with
JavaScript

JS

JavaScript History

Netscape wanted to make the web more Dynamic

making a scripting language

Netscape hired **Brendan Eich** for that reason

Defend Idea by do a prototype

Eich do a prototype in 10 days

JavaScript first name was **mocha**

LIVE

JavaScript Second name was **LiveScript**

And finally be **JavaScript**

JS

Fact #1

“

There are two types of people, One who writes it **“Java Script”** and the other who writes it **“JavaScript”**. First one has no idea about what JavaScript is.

”

JS

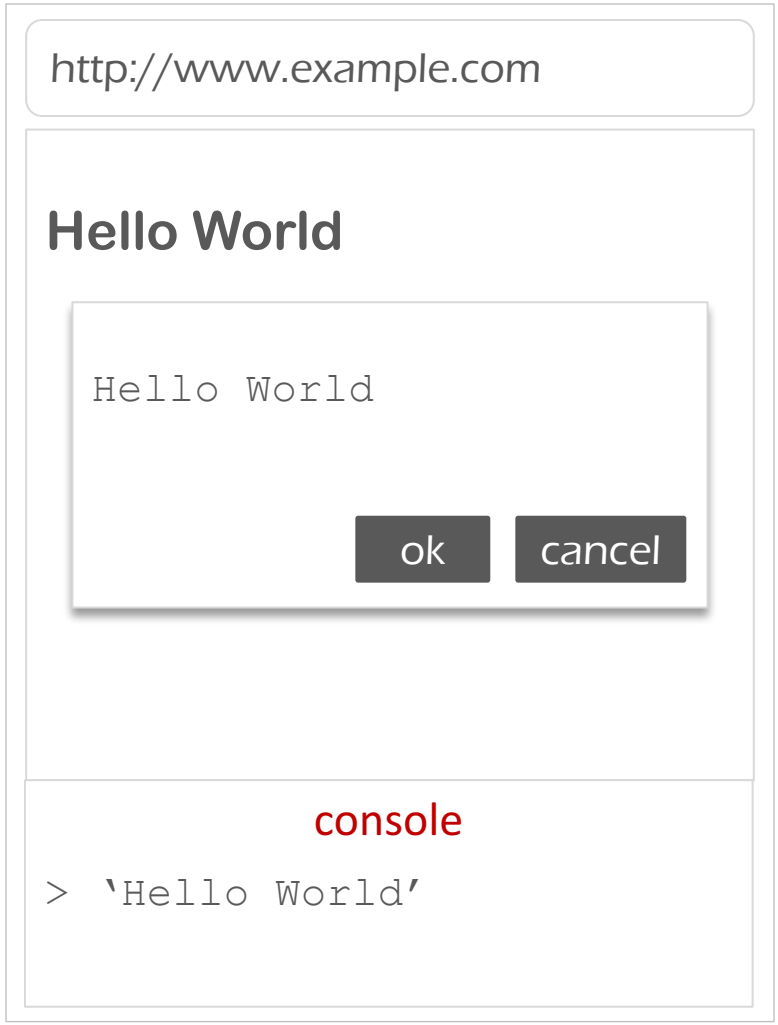
JavaScript Language Core

Hello Word!

```
document.write('Hello World')
```

```
console.log('Hello World')
```

```
alert('Hello World')
```



Where To ?

```
<html>
        <head>
         <script>
                 alert('hello world');
         </script>
         <script  src="myscript.js"></script>
        </head>
         <body>
         <p>Hello</p>

         <script>
                 alert('hello world');
         </script>
         <script  src="script.js"></script>
         </body>
<html>
```

index.html

```
alert('hello world');
```

script.js

JavaScript Syntax

- JavaScript is case sensitive

Var **is not equal to** **var**

- JavaScript statements are separated by **semicolons** (;).
- Variable Names follows this rules:
  - the first character must be a letter, an underscore (_), or a dollar sign ($).

$dollar **(√)** _underScore **(√)** name **(√)** 12twelve **(x)**

  - Subsequent characters may be letters, digits, underscores, or dollar signs.

$do22ar twelve12

Declaring Variables

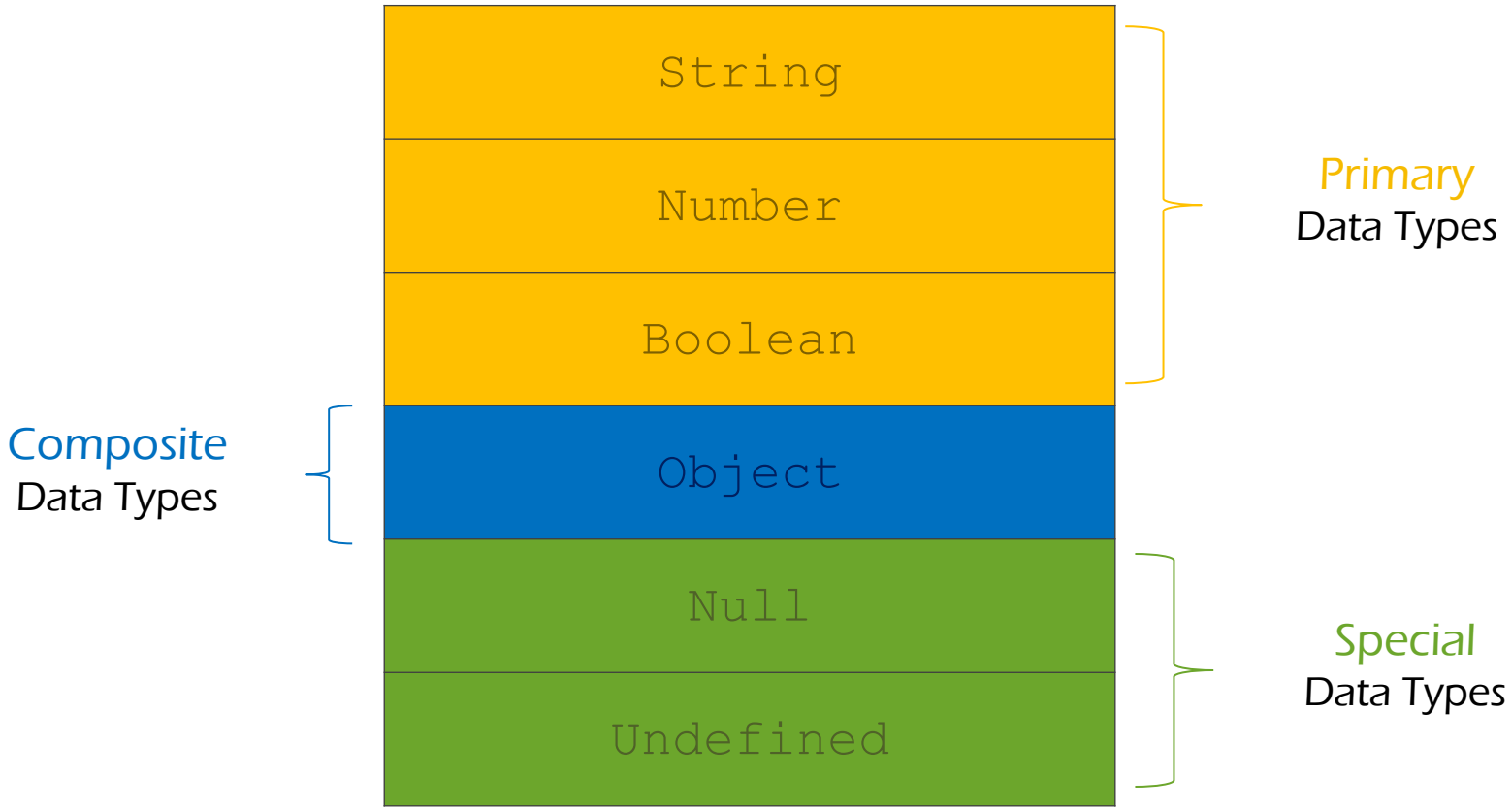
```
var name;

var name, age, email;

var name, age=12;
```

JS

Data Types


String
Number
Boolean
Object
Null
Undefined
Primary
Data Types
Composite
Data Types
Special
Data Types

Primary Data Types

String

Any character array or text quoted

```
var str1="hello JS";

var str2='11.26';

var str3='false';
```

Number

Any Numeric value but **not** quoted

```
var num1= 8;

var num2= 11.26;
```

Boolean

Has only two values true or false

```
var isBool= true;

var isStr= false;
```

JS

Special Data Types

null

This describes the no valid value ,
And has only one value null

```
var thisIsNull = null;
```

undefined

The undefined value is returned when you declare a variable that has never had a value assigned to it.

```
var num1; //num1 is undefined
```

JS

JavaScript is Dynamic

```
var var1;                          //its type is undefined

var1 =“ahmed”;                     //Now, its type is string

var1 =12;                          //Now, its type is number

var1 = true;                       //Now, its type is boolean
```

Checking variables data types

```
typeof variable_name
```

```
var name;

typeof name; //undefined

name ="ahmed";

typeof name; //string

name = null;

typeof name;  //object How??

typeof name == 'object'; //true
```

JS

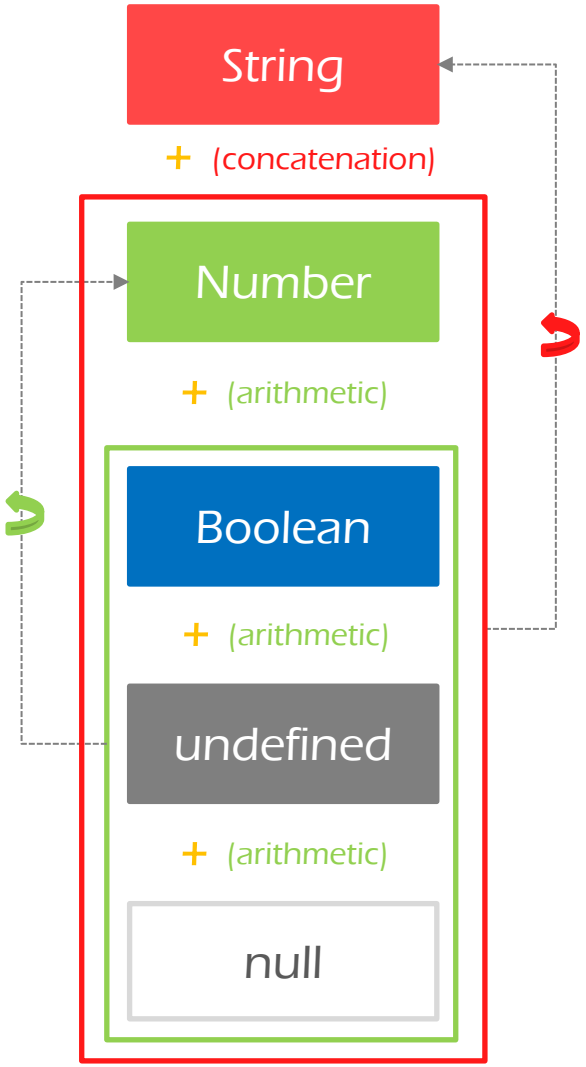
Operators

Arithmetic Operators

Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y

JS

+ operator



input	output
`"ITI" + " OS"`	ITI OS
`20 + "17"`	2017
`"is Exist " + true`	is Exist true
`"not " + undefined`	not undefined
`37 + null`	37
`37 + undefined`	NaN
`true + false`	1
`true + undefined`	NaN

Comparison Operators

a <op> **b**

Operator	Description
==	Return true if value of **a** *equal* to value of **b**.
===	Return true if value and type of **a** *equal* to value and type of **b**.
!=	Return true if value of **a** *not equal* to value of **b**.
!==	Return true if value and type of **a** *not equal* to value and type of **b**.
>	*greater than*
<	*less than*
>=	*greater than or equal to*
<=	*less than or equal to*

JS

Fight No#1

JS

==

It compares only the variable value

===

It compares the variable type and value

== Operator

1 | ● == ● ------> ● === ●

2 |  ==  ------> Number() === Number()

3 |  == null or undefined ------> false

4 | null == undefined ------> true

input	output
"20" == 20	true
0 == null	false
"true" == true	false
NaN == NaN	false
undefined == null	true

input	output
true == 1	true
true == 4	false
false == 0	true
NaN == undefined	false

Logical Operators

&& and Gate

|| or Gate

! not Gate

input	output
`(0 == null) && (true == 1)`	`false`
`(0 == "") || ("2" == 2)`	`true`
`!(null == undefined)`	`false`

JS

&& Operator

&& operator seeks for **falsy** value and return the **first truthy value** it find or the **last value** it stops at

input	output
`true && 4`	`4`
`0 && true`	`0`
`1 && 2`	`2`
`'Ahmed' && ''`	`''`
`false && null`	`false`

JS

|| Operator

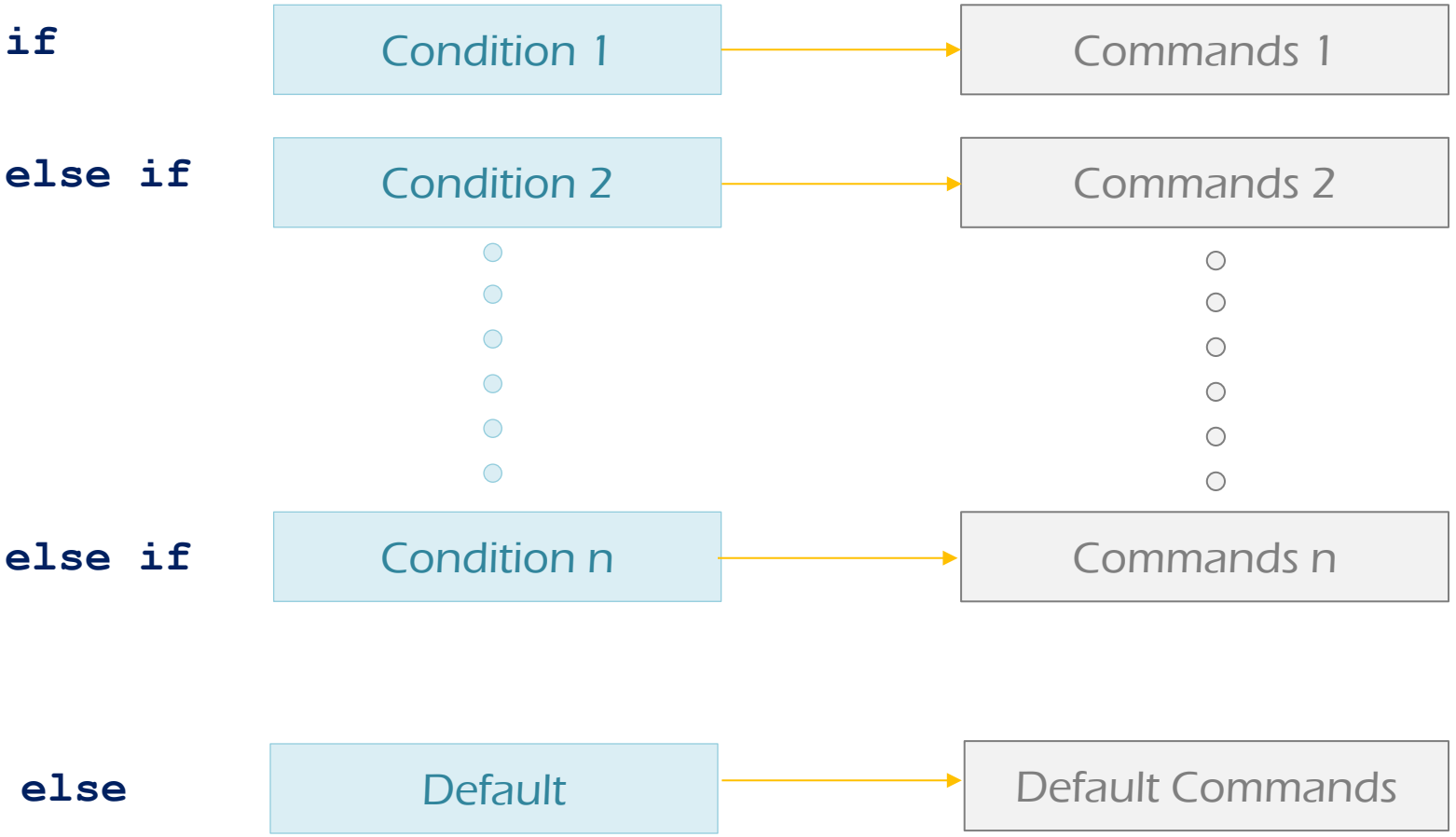
|| operator seeks for truthy value and return the first truthy value it find or last value it stops at

input	output
`true || 4`	`true`
`0 || true`	`true`
`1 || 2`	`1`
`'Ahmed' || ''`	`'Ahmed'`
`false || null`	`null`

JS

Control Flow

If statement



Falsy Values

```
if (name) {
        alert("hi");
}else{
        alert("Bye");
}
```

If name has falsy value it will execute the code in the Else statement

So what is the **falsy values** :

0 , false, null ,undefined, "", NaN

JS

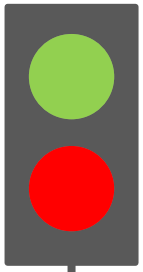
switch...case

```
switch (typeof typedVar) {
    case 'boolean':
        console.log('blue')
        break
    case 'number':
        console.log('green')
        break
    case 'string':
        console.log('red')
        break
    default:
        console.log('grey')
        break
}
```


```
var typedVar = 3
```

```
> green
> red
> grey
```

while Loop



Condition:

```
while (Green ===  true && Red === false)
{
```

Command:

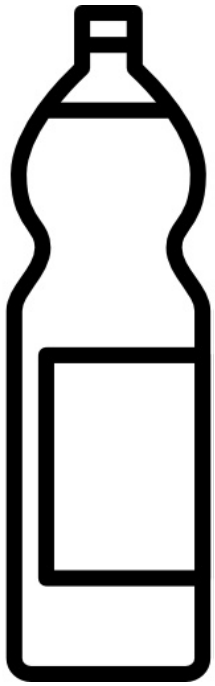
```
  Car.move();
}
```

for Loop

```
for(var i=0;i<5;i++){

    Bottle.fill(cup,water);

}
```



i = 0 ≤ 5

For loop will finish executing here

Fight No#2

JS

`continue`

It makes program skip the current iteration of loop without completing it

`break`

It makes program exit loop without completing the remaining iterations

Dialogs

Alert Dialog

```
alert(text);
```

Return : Doesn't Return any value

```
alert("Hello JavaScript!");
```

OR

```
var greetings = "Hello JavaScript!";
alert(greetings);
```

JS

Prompt Dialog

```
prompt(text, default return value);
```

Return : String

```
var person = prompt("Please enter your name", "Ahmed");

console.log(person) //person = Ahmed
```

JS

Confirm Dialog

`confirm(message)`

Return : Boolean

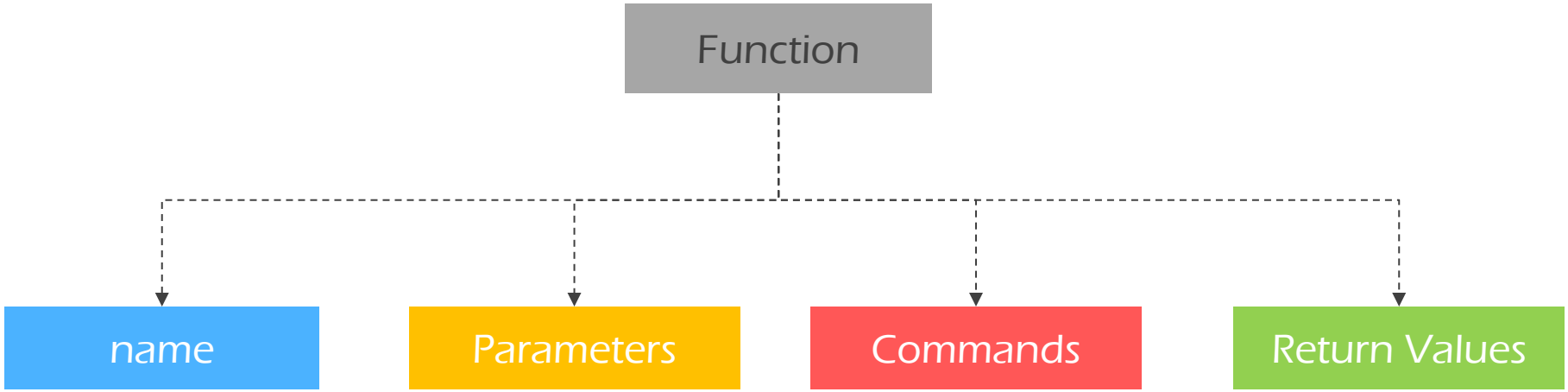
```
var isReady = confirm("Are you ready?");

if(isReady){
        alert("Yes");
}else{
        alert("No");
}
```

JS

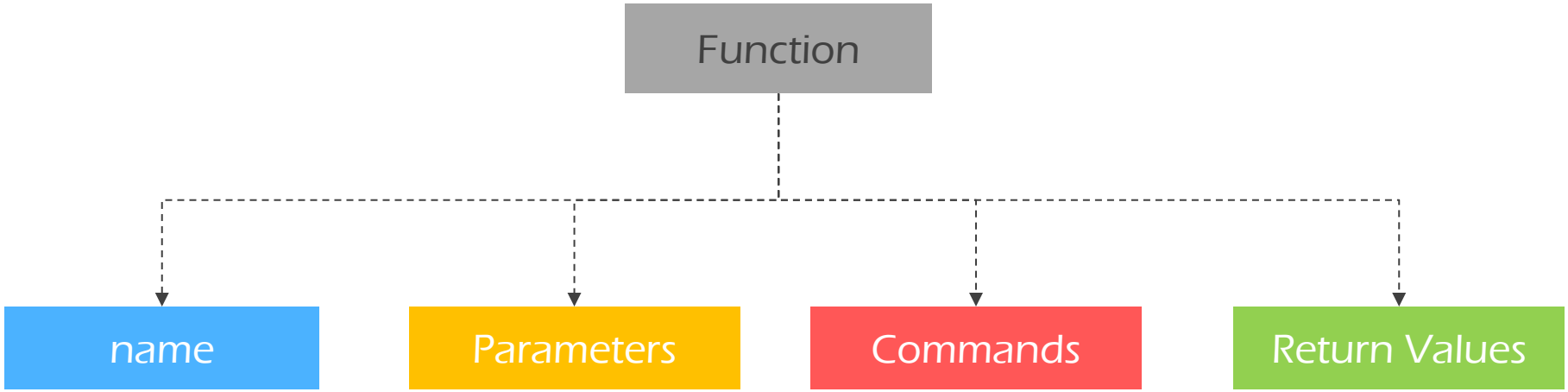
Functions

Intro




```
function name(parameter1, parameter2, parameter3)
{
    code to be executed
    return true;
}
```

Intro




```
function multiply(num1,num2) {
    var result = num1 * num2;
    return result;
}
```

Calling it:

```
var result = multiply(3,4);
alert(result); // result = 12
```

Scope

Global Scope

```
var  globalVar = 0;
```

function1

```
{
 var funcOneVar = 1;
 globalVar++;
 console.log(globalVar);
 console.log(funcTwoVar);
}
```

function2

```
{
 var funcTwoVar = 4;
 globalVar++;
 console.log(funcTwoVar);
 console.log(funcOneVar);
}
```

```
function1();

function2();

console.log(globalVar);

console.log(funcOneVar);
```

Result

```
1
undefined
4
undefined
2
undefined
```

Objects

Everything in JavaScript is an Object

JS

Intro

Bike is an Object

It has characteristics

Properties

```
Bike.color
Bike.height
Bike.width
Bike.weight
```

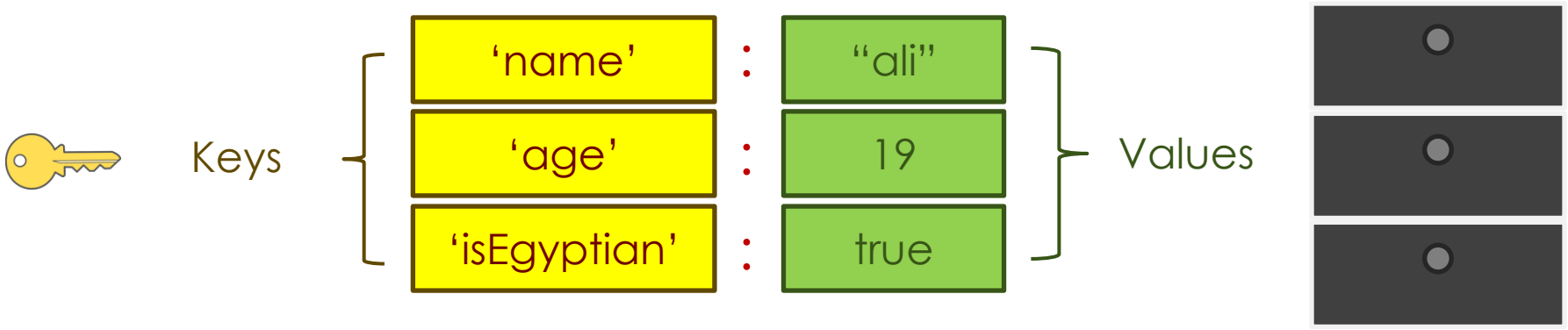
```
Bike.move()
Bike.stop()
Bike.brake()
Bike.jump()
```

Methods

It does some actions

Literal Object

```
{
        name: 'ali',
        age: 19,
        isEgyptian: true
}
```

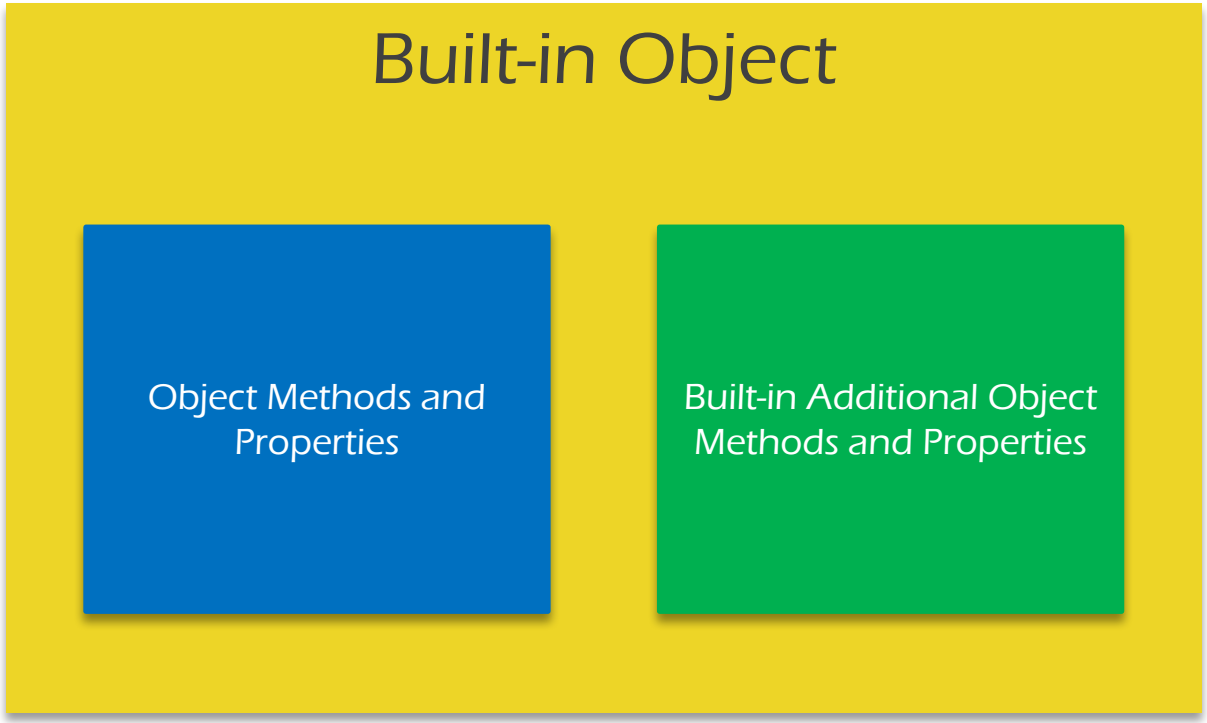


That's Enough for now !

Built-in Objects

overview

They are helper objects that wrap some methods and properties about something like **Date**, **Mathematical Operations**, etc.



Strings

```
var  message = "this is string"
```

input	output
`message.toUpperCase()`	`THIS IS STRING`
`message.slice(5,7)`	`is`
`message.replace("is", "was")`	`thwas is string`
`message.charAt(2)`	`i`
`message.indexOf("is")`	`2`
`message.lastIndexOf("is")`	`5`

Numbers

```
var  num = 15.528
```

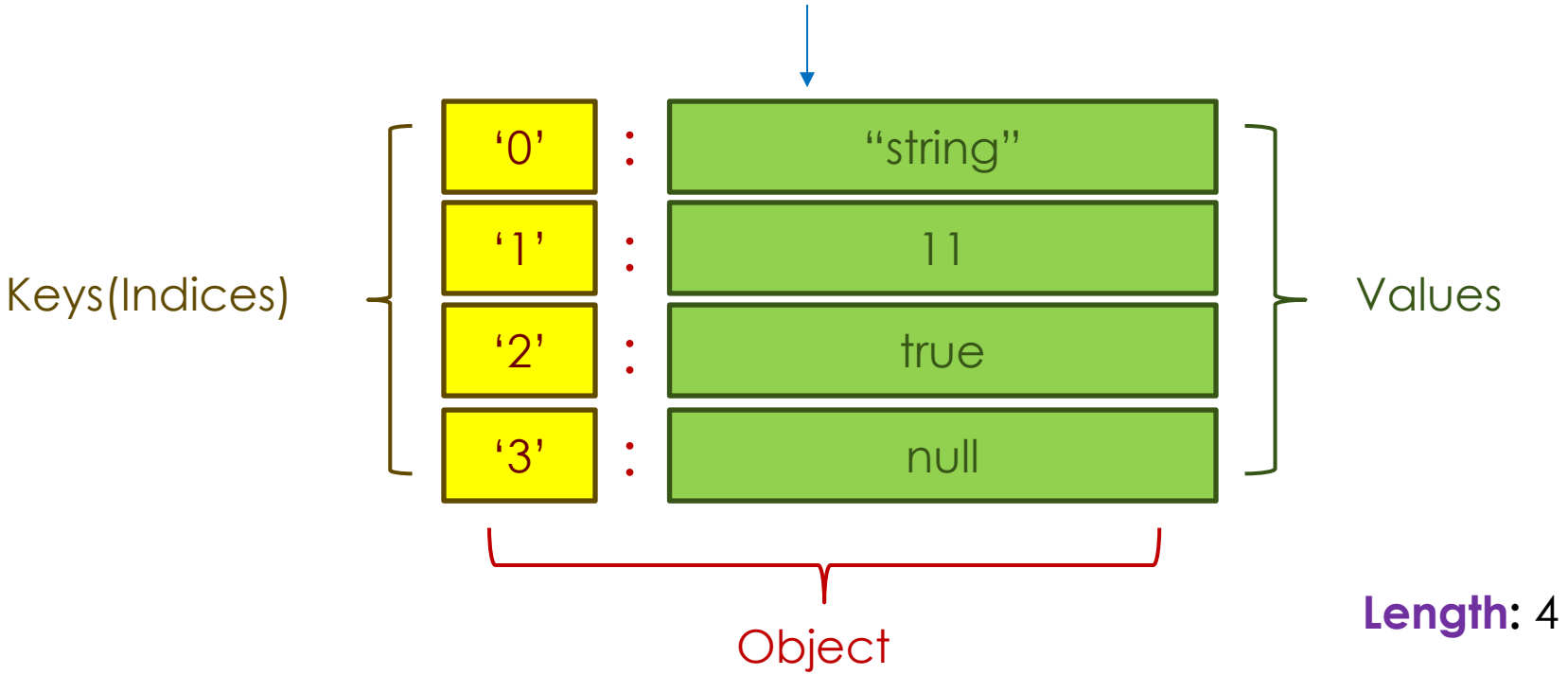
input	output
`num.toString()`	`"15.528"`
`num.toFixed(2)`	`"15.53"`
`num.toPrecision(3)`	`"15.5"`
`num.toPrecision(2)`	`"16"`
`parseInt(num)`	`15`

Arrays

Arrays are a special kind of objects, with numbered indexes

overview

```
var  myArr = ["string", 11, true, null];
```



Length: 4

```
myArr[50] = "no50";
```

Length: 51

Methods

```
var  myArr = ["C", "JavaScript", "Python", "Java", "php"];
```

myArr

- C++
- JavaScript
- Python
- Java
- go

input	output
`myArr.pop()`	php
`myArr.push("go")`	4
`myArr.shift()`	C
`myArr.unshift("C++")`	0
`myArr.splice(4,0,"Scala")`	[]
`delete myArr[3]`	**true**

JS

Math

The Math object allows you to perform **mathematical** tasks

input	output
`Math.PI`	3.14
`Math.sqrt(25)`	5
`Math.abs(-1)`	1
`Math.floor(1.6)`	1
`Math.ceil(1.4)`	2
`Math.round(1.5)`	2

JS

```
var  d = new Date()
```

MISSION #1

Try

exploring its Methods and Properties

Tips and Tricks

for … in

for…in *used to loop into* object *by iterating its* keys

```
var obj = {
 name: 'Ahmed',
 age: 19
}
var info = ''
for (var k in obj) {
        info += 'My ' + k + ' is ' + obj[k] + ' '
}
// info = 'My name is Ahmed My age is 19 '
```

JS

? operator

```
condition ? success_expression : fail_expression
```

```
var canFly = true
var bird = canFly ? 'Dove' : 'Penguin'
// bird is Dove
```

JS

Challenges

Rules

1. If you have Syntax Error, Solve it yourself. You are able to do that.

2. Mentors exist to guide you *to the best way to solve the problem and why errors raised* not *to solve the problem or trace your code to solve syntax errors*.

3. Steps of Solving the problem:
   - Think.
   - Think again.
   - Use Pen and Papers to convert your thoughts into Procedures.
   - Convert your previous pseudo code into JavaScript Code using its syntax rules.
   - Don't be afraid of syntax errors. It is easy to solve. Read it clearly and you will solve it.
   - Check the output of every step you do and then check them all.

4. The most important rule is to enjoy challenging yourself and don't stress your mind by the headache of assignments delivery's deadlines.

JS

Beginner

Fizz Buzz Game

★ ☆ ☆

Write a function that take a number ad check if the given number is divided by 3 only, 5 only or both and print the suitable sentence. Follow the below Rule.

Input

Number

15

Output

Sentence

“fizz buzz”

Notes

Rule :

divided by **3 only** =**“fizz”**, divided by **5 only** =**“buzz”**, divided by **3 & 5**= **“fizz buzz”** ,
Neither divided by **3 nor 5** = **“none”**

Lab

JS

Beginner

Bottle Game

★ ☆ ☆

Write a function that take an array of persons' names and return two random names of them.

Input

array

["ahmed", "islam", "sandra", "Fatma", "Ali"]

Output

array

["sandra", "Ali"]

Lab

JS

Intermediate

Character Game

★ ★ ☆

Write a function that take a sentence and a letter to search for it in the given sentence and return its locations in that sentence.

	String	Letter
Input	“This is javaScript”	“i”

	array
Output	[2, 5, 15]

Lab

JS

Advanced

Greedy Game

★ ★ ★

Write a function that take a number and follow the below rule to convert it into dollars, quarters, dime, nickels and cents.

Input

Number

15.92

Output

Sentence

You have 15 dollar, 3 quarter, 1 dime, 1 nickel and 2 cent

Notes

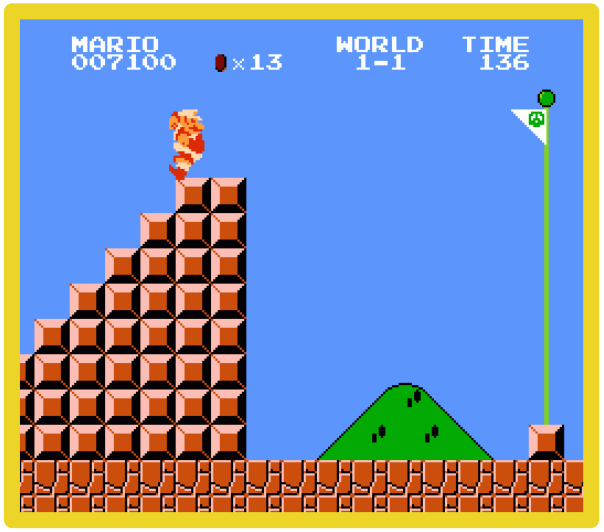
Rule :
1 dollar = 100 cent, 1 quarter = 25 cent, 1 dime = 10 cent, 1 nickel = 5 cent

Lab

JS

Bonus

Mario Game



Input

Number

5

Output

Sentence

```
*
**
***
****
*****
```

Lab

Bonus

Who Am I Game

And Where am I ?

Do you fly?

Yes

Are You Wild?

Yes

Eagle

No

Parrot

No

Do you live under sea?

Yes

Are You Wild?

Yes

Shark

No

Dolphin

No

Are You Wild?

Yes

No

Cat

Lab

Thank You

ahmedmowd@gmail.com

/in/ahmedmoawad

/AhmedMoawad

/ahmedmaawad