# History of JavaScript

History:- In 1995, A Netscape (browser) programmer named Brandan Eich developed a scripting language in just 10 days.

Originally name (first name):- Mocha
Second name:- LiveScript

At that time java is famous programming language. So, for marketing purpose LiveScript changed into javascript.

* Java and JavaScript both are different programming language. nothing is common.

Mocha → LiveScript → JavaScript

In 1997, there is another famous browser that was internet Explorer (Microsoft browser).
Then, Microsoft copied javascript features made own language named as Jscript.
In Browser war (Netscape vs internet explorer)

Netscape → JavaScript
Internet Explorer → Jscript

EcmaScript is born....

**Ecma International :-** Ecma international is an industry association founded in 1996, dedicated to the standardization of information and communication systems.

JavaScript + Ecma → EcmaScript.
(Rules)

Problem solved :- We can implement scripting language for different browser. (same)

First EcmaScript.

ES1 → 1997

ES5 → 2009 (lots of new features)

ES6 (ES 2015) → 2015 (Biggest update for JS)

ES6 is also known as Modern JavaScript

Ecma have a technical community known as TC39 had decided thatafter 2015. We release javascript with new features every year (Annual release).

@programmer-girl--

# JavaScript Features.

**Features:-**

Case sensitive
Dynamically typed
Cross-platform
Interpreted
Object-oriented Scripting language
Backward compatible

@programmer-girl--

# JavaScript variables

**Variables:-** variables stores the data which can be changed or used when we need. There are there *keywords to declare a variable.

keywords are the predefined words in programming languages.

- var  var name = 10;
- let  let name = 10;
- const  const pi = 3.14;

# Datatype in JavaScript

There are two types of Data

1. Primitive
2. Non-primitive.

Primitive datatypes are :-

- Number
- Null
- String
- Bool
- Undefined
- Bigint
- Symbol



Non-Primitive datatypes are:-

- Array
- object
- RegExp.

# JavaScript Hacks

1. Convert string to number.

Put the pulse (+) before the string

For Example:-

```
let str = "9";
console.log(typeof(+str));
```

2. Convert numbers into string

Add a empty string with the numbers

For Example:-

```
let num = 10;
console.log(typeof(num + ""));
```

@programmer-girl--

# JavaScript String.

**String:-** String are used to store textual form of data like word, sentence. It follows zero based indexing.

```
let str = "pro";
let str = 'pro';
let str = `pro`;
```

## JavaScript String Method

| | |
|---|---|
| trim() | slice() |
| charAt() | tostring() |
| concat() | Substring() |
| indexOf() | toUpperCase() |
| lastIndexOf() | tolowerCase() |

@programmer-girl--

# Undefined in Javascript

- Accessing an uninitialized variable returns undefined.

```
let str;
console.log(str); //undefined
```

- Accessing a non-existing property of an object returns undefined.
- Accessing a out-of-bounds array element returns undefined

# Null in JavaScript

- null means 'no value' assign to variable.
- typeof null returns 'object'
- Null is treated as false value.

@programmer-girl--

# JavaScript BigInt

**BigInt:-** BigInt is a primitive Datatype which is used for large numeric values it doesn't represent decimal values.
It is used to represent values greater than 2^53-1.

## Declaration of BigInt

- By appending n at the end of numeric values.

```
var num= 9876543219865252772n;
```

- By passing the values as an argument to the BigInt().

```
var num= BigInt(98765432198652527 7);
```

@programmer-girl--

JS

# Ternary Operator

**Ternary Operator:-** It is also called **conditional** operator.

→ It takes three operands.

→ It makes the code more concise.

@programmer-girl--

**Syntax:-**

```
let variableName = condition ? True : False;
```

If the condition is true expression after ? will executes. If it is false, expression after : (colon) will executes.

@programmer-girl--

**For Example:-**

```
let age = 18;
let warning;
age >= 18 ? (warning = "You can play")
          : (warning = "You cannot play");
console.log(warning);
```

**output:-** You can Play.

# JS Boolean Data Type

**Boolean** :- It can hold only two values: true and false.

For Example:-

```
Var Read = true;  ] - typeof(Read) ]
Var Eat = False;                   Boolean
```

Boolean values also come as a result of comparisons.

For Example:- @programmer-girl--

```
Var x = 1, b = 4, y = 8;
Console.log (b>x)  //output:- true
Console.log(b>y)   //output:- false
```

== and ===

== (Double equals operator) :- Known as the **Equality** or **abstract** comparison operator.

→ It compare variables, ignores datatype.

=== (Triple equals operator) :- Known as the **identity** or **Strict** comparison operator.

→ It compare variables as well datatype.

# JS Truthy and Falsy Values

**Truthy values:-** It is a value that is considered true when encountered in a Boolean Context.

Example :-

true, {}, [], 42, "0", "False",
new Date(), -42, 12n, 3.14, -3.14,
Infinity, -Infinity.

@programmes-girl--

**Falsy values:-** It is a value that is considered false when encountered in a Boolean context.

Example :-

Undefined, null, NaN, false, "",
0, -0, on (BigInt)

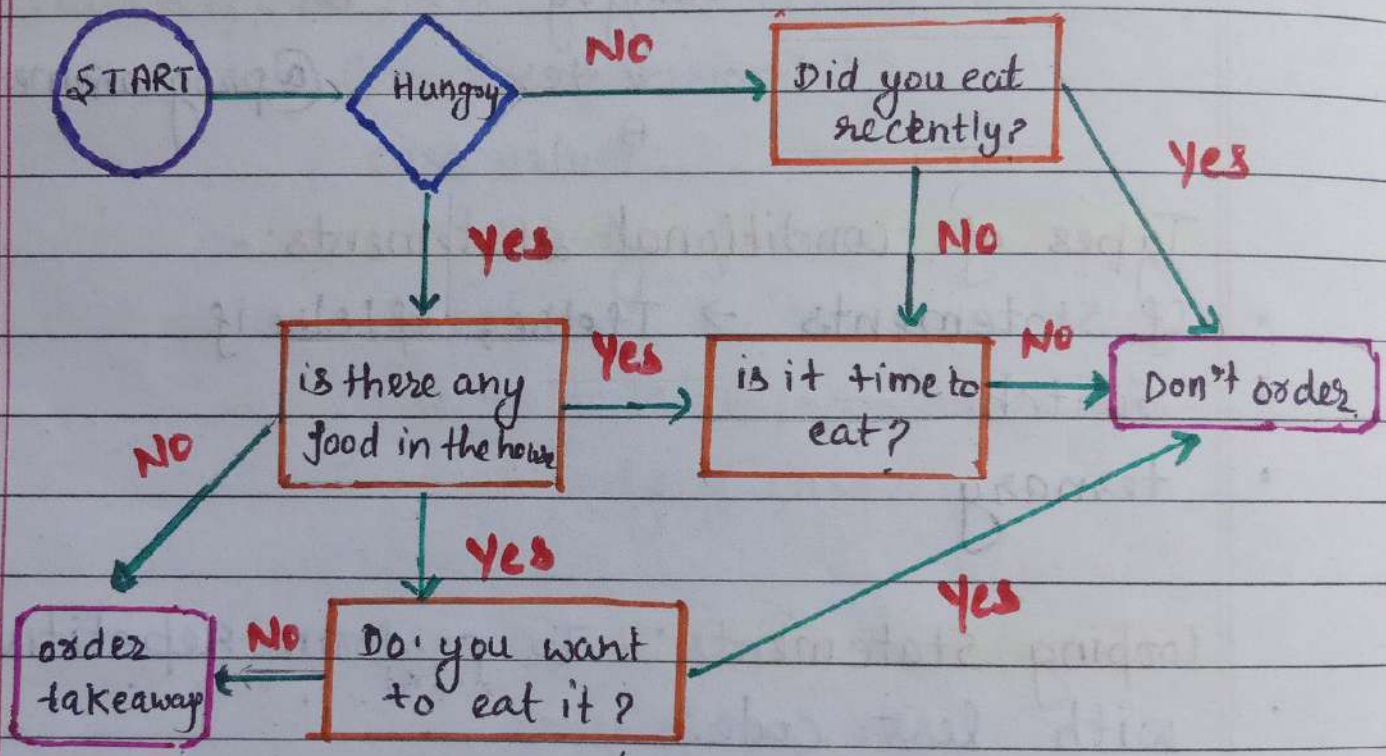
```
Var values = 42;
if (values) {
    console.log (true);
}
else {
    console.log (false);
}
```

output :- true

# Control Flow

**Control Flow:-** It allows our program to make decisions about what code is executed and when.

**For Example:-**

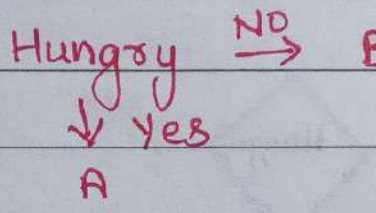



@programmer-girl.

Control Flow have two types of Statements

1. Conditional Statements.
2. Looping Statements.

**Conditional statements:-** Conditional statements are basically checks to see if a certain condition is either true or false. If the condition is true then run code A, if it's false then rund code B.




**Types of conditional statements:-**

- If statements → If else, if/else if
- Switch
- ternary

**Looping Statements:-** To perform repetitive task with less code.

**Types of loops:-**

- for loop
- do/while
- for .. in
- for .. of.

# Switch Statement

**Switch statement :-** It evaluates an expression compare its result with case values and execute the statement associated with the matching case.

**Switch Syntax :-**

```
Switch (expression) {
 case value1:
      // body of case 1.
  break;
 case value2:  @programmer-girl--
      // body of case 2
  break;
 default;
      // body of default
}
```

**break :-** It is optional. It is used to end the ~~stat~~ switch statement.

**Default :-** If there is no matching case, the default body executes. It is optional.

# For loop

**For loop:-** For loop executes a block of code as long as a specified condition is true.

Syntax:-



```
for (initializer; condition; iterator){
  {
      // statements
  }
```

→ **Initializer:-** It is an expression that initialize the loop, it executed once.

→ **Condition:-** It is a boolean expression that determines whether the for loop should execute or stop.

→ **Iterator:-** For statement executes the iterator after each iteration.

→ **Example:-**

```
for (let i = 2 ; i < 4 ; i++) {
  Console.log(i); }
```

# While, Do While loops

**While loop :-** While loop executes statements as long as the conditions are true. If the condition become false, the loop is terminated.

**Syntax :-**

```
while (condition) {
    // statements
}
```

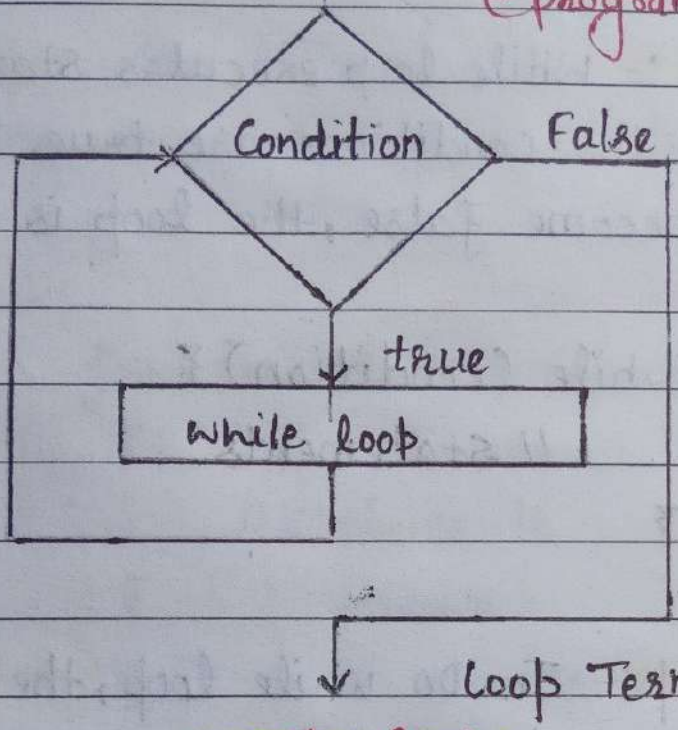
**Do While loop :-** In Do while loop, the block of code executed once even before checking the condition.

**Syntax :-**

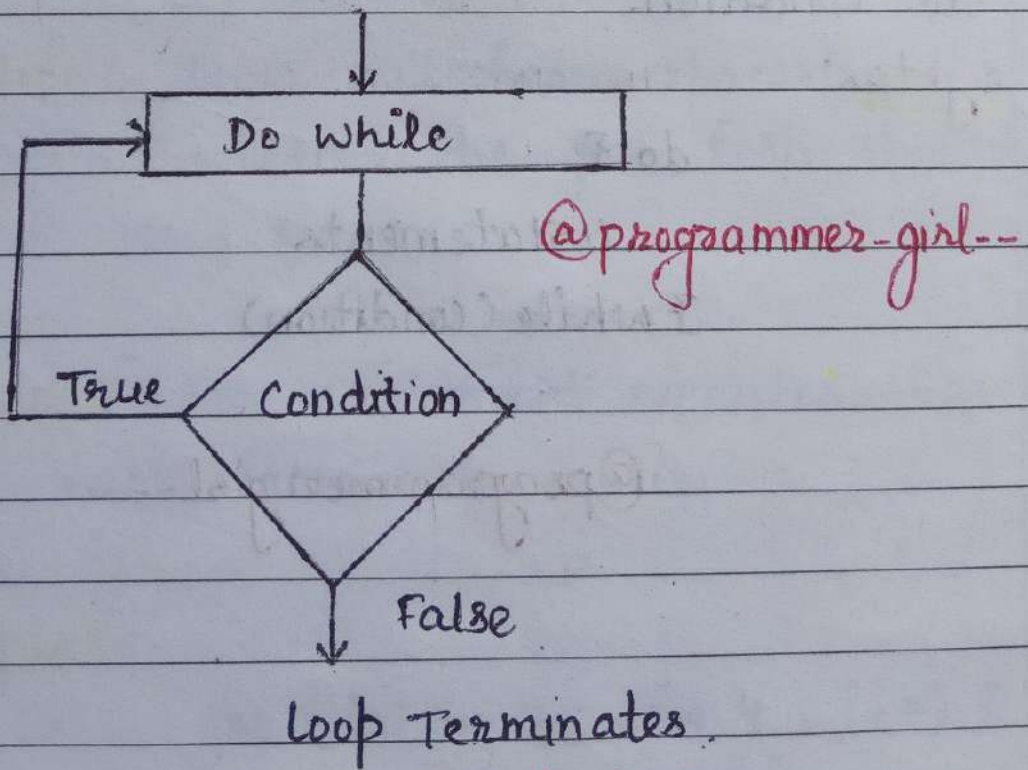
```
do {
    //statements
} while (condition)
```

@programmer-girl--

@programmer-girl--




-: While loop :-




-: Do... While loop :-

# JavaScript Functions

**Function:-** A function is a block of code that performs a specific task.

**Declare Function:-**

```
function funName (){
    //statements
}
```

function is declared using the function keyword.

@programmer-girl--

**Call Function:-**

```
function funName (){
    //statements
}
funName (); → Call Function.
```

**Example:-**

```
function myName (){ → Declare Fun.
    console.log ("Smily");
}
myName (); → Function call
```

Output:- Smily

## Advantage of function:-

Reusability
less code
Easy to understand

Function Parameters:- When we declare function we specify the parameters.

Function Arguments:- When we call function we specify the arguments.

For Example:-

```
Function example (Parameter){
 console.log (Parameter);
}
let argument = 'arg';
example (argument);
```

@programmer-girl--

# -: Intro to Arrays :-

**Arrays:-** It is a ordered collection of items.

Element / item

```
let pets = ["cat", "dog", "cow"];
Index. →  0      1      2
```

## JavaScript Array Characteristics

1. It can hold values of mixed types.
2. Size of Array is dynamic.

@programmer-girl--

```
let mixed = [1, 2.5, "cat"]; → Mixed Type.
pets.push("Monkey");
console.log(pets);           → Dynamic Size.
```

## Accessing Array Elements

Arrays are zero-based indexed. It means the first element of array starts at index zero.

```
let pets = ["cat", "dog"];
console.log(pets[0]); → Accessing element
```

**output:-** cat