For interactivity in the website, we use JavaScript

Also called JavaScript ES6 (Ecma Scripts 6).

For popup command:

alert(“Hello”);

Data Types

‘Hello’ – String

1, 2, 3 – Integer

Boolean: describe either data is true or false

Variable Syntax

var (name of variable) = variable

prompt();

Asks user to input data

Concatenation of Strings

Ex.

var myName= 'Mehul';

var yourName = prompt("Your name is: ");

alert('Hi, '+ yourName+ '\nMy name is '+ myName);

Naming Conventions

Give meaningful names

Can’t begin with a number

Can’t contain space

Can only have alphabets numbers and $ and \_

* camelCase (used in JS)
* PascalCase (used in JS)
* snake\_case

Functions

(variable\_name).length() counts no. of characters

(variable\_name).slice(x,y) gets the no of characters between x and y where x and y are numbers and the first character begins at 0

(variable\_name).toUpperCase() converts string to upper case

Or toLowerCase

Operators

+, -, /, \* or %(modulo)

Increment expressions

Ex

Var x =5;

X = x + 1 or x++

=6

Or

Var x =5;

X = x-1 or x—

=4

For changing value more than 1

Var x =5;

Var y = 3;

X +=y; //8

Functions

Create function

**Normal function**

function function\_name(){

}

**Parameters**

Function function\_name(parameter){

}

Eg.

function getMilk(money) {

console.log("leaveHouse");

console.log("moveRight");

console.log("moveRight");

console.log("moveUp");

console.log("moveUp");

console.log("moveUp");

console.log("moveUp");

console.log("moveRight");

console.log("moveRight");

var bottles = Math.floor(money/1.5);

console.log("Buy " + bottles + " bottles of milk");

console.log("moveLeft");

console.log("moveLeft");

console.log("moveDown");

console.log("moveDown");

console.log("moveDown");

console.log("moveDown");

console.log("moveLeft");

console.log("moveLeft");

console.log("enterHouse");

alert("Buy " + bottles + " bottles of milk");

}

var cost = prompt("Enter the amount: ");

getMilk(cost);

Callling function

Function\_name();

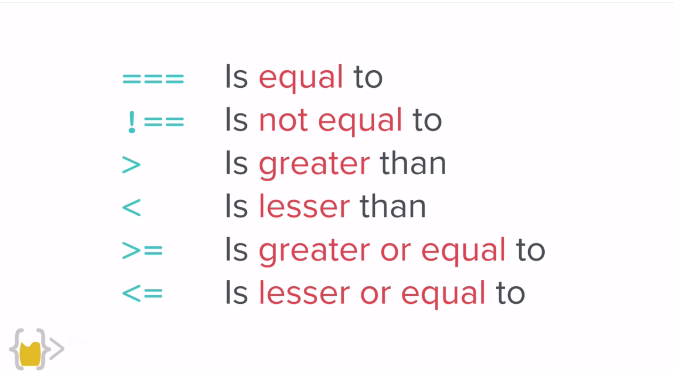
Console.log

For calling instructions in JavaScript console

Wont be able to see by end user

Various math methods

Comparators and Equality



Difference b/w == and ===

== doesn’t check for data types

=== checks for data types

Combining Comparators

Operator 1 && Operator 2 for both conditions to be true

Operator 1 || Operator 2 for either to be true

! for false

Array

Collection of data

Ex

Var eggs = [data1, data2, data3, data4]

Eggs[0] returns data1

Var myegg = eggs[1] assigns “data 2” to myegg

If Elseif Else

If (condition){

} else if (condition) {

} else (condition){

}

Loops

While (condition){  
}

JavaScript implementation in website

Inline JavaScript

Avoid it, not great, not modular

<tag\_name onload=”(JavaScript);” >

Internal JavaScript

<script type=”text/JavaScript”>

JavaScript goes here;

</script>

External JavaScript

<script src=”path.js” charset=”utf-8”> </script>

Placing JavaScript matters. We place JavaScript where we want it to run

If we want to change some element using JavaScript, we must first have the existing tag

Document Object Model (DOM)

Catalogs the webpage into individual objects which can be manipulated using JavaScript

Analyses the entire webpage and its elements and creates a tree

Ex

document.firstElementChild selects the first html elements that will be head element

document.lastElementChild selects the last element which might be the body

Using document.querySelector(“element\_name/#id/.class”).method\_name we can manipulate it using methods

The element has properties and methods which can be manipulated.

For example, properties can be color, size, etc. and methods can be clicking a button, etc,

Get property

Ex. Car.color; will give us the color of an element

Set property

Car.numberOfDoors = 0;

Call method

Car.drive(); calls a method to drive the car;

Select Elements

On selecting an item, we get always an array and then we have to use index no to select the child element

Document.getElementByTag(‘’)[index\_no\_in\_array] or Document.getElementByClassName(‘’) [index\_no\_in\_array] or ById(‘’)

document.querySelector(“element\_name/#id/.class”).textContent = “text”; can be used to manipulate text of an html element

Adding attributes to html elements

document.querySelector(“element\_name/#id/.class”).attributes; gives all the attributes of the selector

document.querySelector(“element\_name/#id/.class”).getAttribute(“desired\_attribute”); gives the desired attribute

document.querySelector(“element\_name/#id/.class”).setAttribute(“desired\_attribute”); sets to desired attribute

addEventListener(“when\_this\_event\_triggers”, function() {

} );

We can also use higher order functions. In higher order functions, we can use a function as an input and then trigger another function.

For example: this calculator programme.

function add(num1, num2){

return num1 + num2;

}

function subtract(num1, num2){

return num1 - num2;

}

function multiply(num1, num2){

return num1 \* num2;

}

function divide(num1, num2){

return num1/num2;

}

function calculator (num1, num2, operator){

return operator (num1, num2);

}

console.log(calculator);

Here operator will be the local function that shall be carried out.

Objects

Ex.

Var houseKeeper1 = {

name: "Katherine",

yearsofExperience: 12,

}

Constructor Functions:  
So, these functions can be used to create multiple objects with same properties.

Syntax:

function HouseKeeper (name, yearsOfExperience, cleaningRepertoire ) {

this.name = name,

this.yearsofExperience= yearsOfExperience,

this.cleaningRepertoire = cleaningRepertoire,

}

Var HouseKeeper1 = new HouseKeeper(“Katherine”, 12, [“bathrooms”,”lobby”,”bedrooms”]);

Var HouseKeeper2 = new HouseKeeper (“jane”, 11, [“bathrooms”,”balcony”,”bedrooms”]);

And so on.