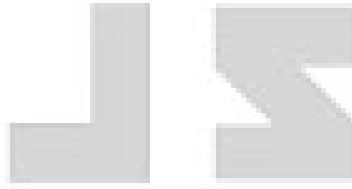


CHAPTER 4 – OBJECTS & ARRAYS



JS Objects

JS Objects are the variables that can contain many values. The values are written as name:value pairs.

e.g.- *const car = {type:"Fiat", model:"500",color:"white"};*

↙ ↘
name : value

You can access object properties in two ways:

1. `objectName.propertyName` → `car.type`;
2. `objectName["propertyName"]` → `car["type"]`

JS Array

An array can hold many values under a single name, and you can access the values by referring to an index number.

e.g.- *const cars = ["Audi","Volvo","BMW"];*

Note : Array indexes start with 0. [0] is the first element. [1] is the second element.

JS Array Properties & Methods

1. The Length property → Returns the length of the array.

e.g.- *let length = cars.length;*

2. Pushing Array Elements → Adds a new element in the array.

e.g.- *cars.push("Nissan");*

3. Popping Array Elements → Remove the last element of the array.

e.g.- *car.pop();*

4. Deleting Array Elements → Delete an element from the array.

e.g.- *delete cars[0];*

5. Merging Arrays → Merges two or more arrays.

e.g.- i - *const myArr=myArr1.concat(myArr2);*

ii - *const myArr=myArr1.concat(myArr2,myArr3);*

6. Array Sorting → Sorts the array.

e.g.- *cars.sort();*

7. splice() → Used to add new items in the Array.

e.g.- *const fruits ["Apple","Mango"];*

fruits.splice(1,2, "Lemon","Kiwi");

8. slice() → Used for slicing the Array.

e.g.- *const fruits = ["Banana","Orange","Lemon"];*

const citrus = fruits.slice(1,3);

9. `toString()` → Converting Arrays to Strings.

e.g.- `const fruits = ["mango", "apple", "pear"]`
`document.getElementById("demo").innerHTML`
`= fruits.toString();`

10. `join()` → Joins the array elements.

e.g.- `const fruits = ["mango", "apple", "pear"]`
`document.getElementById("demo").innerHTML`
`= fruits.join(" * ");`

JS Array forEach()

The `forEach()` method calls a function once for each array element.

e.g.- `const number = [45, 4, 9, 16, 25];`
`let txt = "";`
`numbers.forEach(myFunction);`
`function myFunction(value, index, array){`
`txt += value + "
";`
`}`

JS Array map()

The map() method creates a new array by performing a function on each.

e.g.- *const numbers1 = [45, 4, 9, 16, 25];*
const numbers2 = numbers1.map(myFunction);
function myFunction(value, index, array) {
*return value * 2;*
}

JS Array filter()

The filter() method creates a new array with Array elements that passes a test.

e.g.- *const numbers = [45, 4, 9, 16, 25];*
const over18 = numbers.filter(myFunction);
function myFunction(value, index, array) {
return value > 18;
}

JS Array reduce()

The `reduce()` method runs a function on array element to produce a single value.

e.g.- *const numbers = [45, 4, 9, 16, 25];*

let sum = numbers.reduce(myFunction);

function myFunction(total, value) {

return total + value;

}