**Cheat Sheet-Java Script**

# Chapter 3: Multiple Values (Arrays)

Single Dimensional Array:

* Arrays are lists of values. These values can be of all data types and one array can even contain different data types. It is often very useful to store multiple values inside one variable.
* Array start with index 0, the first index of an array is always 0.
* Arrays can be grown dynamically in JavaScript, although it will be creating confusions later in the code.
* Array is an object, so couple of properties and functions are associated with an array.
* Values in an Array can be retrieved via Index Values, Syntax is arr[index\_value]. If the index value doesn’t exist, Undefined is returned.
* Common Properties and Methods are
  + Length: Total number of element spaces in an array.
  + Push: Push a value at the end of an array
  + Pop: Retrieves a value from the end of an array and removes it from the array index
    - Push and Pop Methods made array to behave like a Stack Data structure
    - Stacks are Last in First Out (LIFO) Data Structures.
  + Splice: Splice is used to extract specific no of elements from an array and create a new array from original array. The splice also removes the elements extracted in the new array from original.
  + Shift: Used to remove the first element of the array
  + Find: use to find an element in the array
  + IndexOf: to get the index of a particular value in array
  + Sort:
    - For Sorting Numeric Arrays (based on numeric value order)
    - For Sorting String Arrays (based on Alphabetic value order)
* Concat: You can add 2 arrays by using concat, it will return a new array combining all elements from array 1 and array 2. You can also use + to concatenate 2 arrays, although the last element of first array and first element of 2nd array will be concatenate is sort of string. So adviced to use concat instead of +.

Multi-Dimensional Arrays

Earlier, we established already that arrays can contain any data type. This means that arrays can also contain other arrays (which, in turn, can contain… other arrays!). This is called a multidimensional array. It sounds complicated, but it is just an array of arrays: a list of lists:

How a multi-dimensional array works in programming environment

Here is a 2-Dimensional Array.

//MultiDimensional Arrays, Arrays of Array, List of Arrays

let arr1=[1,2,3];

let arr2=[3,4,5];

let arr3=[6,7,8];

//Create an array of arr

let mda=[arr1,arr2,arr3];

console.log(mda);//will show a matrix like structure for 3x3

//Reading an Array

// we have 3 internal arrays with 3 values each in mda. making it a 3x3 matrix

console.log(mda[0][2]);//the first square bracket contains the index of main array.

* The mda[0] returns the array arr1, which further contains 3 elements.
* The mda[0][1] returns the first value of arr1.

3-Dimensional Arrays:

//3 dimensional  array

let arr3d=[mda,mda,mda];//here i have added the mda 3 time in arr3d

console.log(arr3d);

//in order to get values you can write

console.log(arr3d[0][1][2]);//first bracket retuerns the first mda,

                            //2nd bracket returns the arr2 in first mda,

                            //3rd bracket returns the 3rd element in first mda

The number of bracket required is corresponding with the dimension of the array, if you want to get any element from a 2D array, you need 2 brackets to access the index. If it’s a 3D array, the number of brackets will be 3, and so on.

Objects in JavaScript

What is an Object:

Objects are the things you think about first in designing a program and they are also the units of code that are eventually derived from the process.

All objects possess three basic characteristics -- identity, state and behavior.

* Identity means that each object has its own object identifier and can be differentiated from all other objects. Each object's name, or identity, is unique and distinct from other objects.
* State refers to the properties of an object. For example, values of variables in the object contain data that can be added, changed or deleted.
* Behavior refers to actions that the object can take. For example, one object can respond to another object to carry out software functions.

Some of the things in programming that can be defined as objects include the following:

* variables, which hold values that can be changed;
* data structures, which are specialized formats used to organize and process data;
* functions, which are named procedures that perform a defined task; and
* methods, which are programmed procedures that are defined as components of a parent class and are included in any instance of that class.

Objects can do things and can have things done to them. For example, a function or method object can be programmed to modify the contents of a data structure or variable object.

Let’s create an example from Chapter 3

let dog = {

    dogName: "JavaScript",

    weight: 2.4,

    color: "brown",

    breed: "chihuahua",

    age: 3,

    burglarBiter: true

};//The object written in JavaScript is called JavaScript Object Notion (JSON) and is a defacto-standard for all rest-based programming

//lets access some values from the dog object and see in the console

console.log(dog);//Prints how a dog object looks in JSON

console.log(dog.color);//Get some value from dog object.

console.log(dog["color"]);//works in the same manner as line 12

You can update the value in the same manner you are getting the values from and object and assign it a value using assignment operator

dog.color="black";//assigns a new value to Dog object

console.log(dog);//Prints how a dog object looks in JSON

console.log(dog.color);//Get some value from dog object.

console.log(dog);//Prints how a dog object looks in JSON

dog["color"]="white";//assigns a new value to Dog object

console.log(dog.color);//Get some value from dog object.

The above example is an example of a simple object written in JavaScript Object Notion (JSON), how ever the real world examples are not as simple as creating a basic dog object.

When working with Arrays, we realize that they can be used in a combination to create more complex data structures. Similar can be done with Objects. Objects can also hold other objects in order to create a more complex object structure.

Lets create a more complex object from the book example.

let company = {

    companyName: "Healthy Candy",

    activity: "food manufacturing",

    address: {

        street: "2nd street",

        number: "123",

        zipcode: "33116",

        city: "Miami",

        state: "Florida"

    },

    yearOfEstablishment: 2021

};

//lets read the company object

console.log(company);

//get the company name

console.log(company.companyName);

//Get the Address of the Company

console.log(company.address);

Here company address returns as another Java Script Object. Since address can contains further element, the Company object designs as a complex object.

console.log(company.address.zipcode);

Get the ZipCode using the dot(.) operator. Dot operator is often used to access internal properties and behavior of an object.

You can also mix Object and Arrays to form more complex structures.

//Objects with Arrays

company = {

    companyName: "Healthy Candy",

    activities: ["food manufacturing","improving kids' health", "manufacturing toys"],

    address: {

        street: "2nd street",

        number: "123",

        zipcode: "33116",

        city: "Miami",

        state: "Florida"

    },

    yearOfEstablishment: 2021

};

console.log(company);

console.log(company.activities[1]);//Get the value at index 1 in company's activitiy array

Arrays of Objects

Since Arrays can hold any type of data, they can be used to create Arrays for a particular object

//Arrays of Object

let addresses = [{

    street: "2nd street",

    number: "123",

    zipcode: "33116",

    city: "Miami",

    state: "Florida"

},

{

    street: "1st West avenue",

    number: "5",

    zipcode: "75001",

    city: "Addison",

    state: "Texas"

}];

console.log(addresses);

console.log(addresses[1]);//gets the 2nd address in the list

console.log(addresses[0].street);//Get the 1st address in the array and print street

console.log(addresses[0]["street"]);//Get the 1st address in the array and print street