JavaScript Arrays II

Topics Covered:

- Multidimensional Array
    - Creating multidimensional Array
    - Accessing multidimensional Array
    - Adding elements to multidimensional Array
    - Removing elements from multidimensional Array
- Referential Array
    - Array as Objects
    - Array Methods
    - Array Properties

Topics in Detail:

Multidimensional Array

- Multidimensional Array is also known as **array of array**.
- JavaScript **does not** have inbuilt **support** for **multi-dimensional arrays**.
- So we need to **create an array inside an array** to make it work as a Multidimensional Array.

Create a multidimensional Array

```
1st, need to define some 1D array
var arr1 = ["ABC", 24, 18000];
var arr2 = ["EFG", 30, 30000];
var arr3 = ["IJK", 28, 41000];
var arr4 = ["EFG", 31, 28000];
var arr5 = ["EFG", 29, 35000];
// "salary" defines like a 1D array but it already contains some 1D array
var salary = [arr1, arr2, arr3, arr4, arr5];
```

OR

```
var salary = [
   ["ABC", 24, 18000],
   ["EFG", 30, 30000],
   ["IJK", 28, 41000],
   ["EFG", 31, 28000],
];
```

Accessing Multidimensional Array elements

- **Simple index-based notation**

```
// This notation access the salary of "ABC" person which is 18000,
// [0] selects 1st row, and [2] selects the 3rd element
// of that 1st row which is 18000
salary[0][2];
```

- **For many iterations**

```
// This loop is for outer array
for (var i = 0, l1 = salary.length; i < l1; i++) {

    // This loop is for inner-arrays
    for (var j = 0, l2 = salary[i].length; j < l2; j++) {

        // Accessing each elements of inner-array
        documents.write( salary[i][j] );
    }
}
```

Adding elements in multidimensional array

There are two ways to add an element in multidimensional array

- Inner Array
- Outer Array

Adding elements to inner Array

- **Square Bracket Notation**

```
salary[3][3] = "India";

// It adds "India" at the 4th index of 4th sub-array,
// If we print the entire 4th sub-array, document.write(salary[3]);
// the output will be :  ["EFG", 31, 28000, "India"]
```

- **push() Method**

```
salary[3].push("India", "Mumbai");

// It add "India" at the 4th index and "Mumbai" at
// 5th index of 4th sub-array
// If we print the entire 4th sub-array,
// document.write(salary[3]);
// The output will be :  ["EFG", 31, 28000, "India", "Mumbai"]
```

Adding elements to outer Array

```
salary.push(["MNO", 29, 33300]);
// This row added after the last row in the "salary" array
```

Removing Elements in Multidimensional Array

- The **pop()** method is used to **remove elements** from the **inner array**.

```
// Remove last element from 4th sub-array
// That is 28000 indexing starts from 0
salary[3].pop();
```

- The **entire inner array** can be **removed** from the outer array with the **pop()** method.

```
// Removes last sub-array
// That is "["EFG", 31, 28000]"
salary.pop();
```

Referential Array

Array as Objects

- Arrays are a **special** type of **object**.
- In Javascript, **typeOf** operator is used to return an **array object**.
- **Numbers** are used to access the **elements of an array**.

```
const person = ["John", "Doe", 46];
document.getElementById("demo").innerHTML = person[0];
```

- **Names** are used to access the **members of an object**.

```
const person = {firstName:"John", lastName:"Doe", age:46};
document.getElementById("demo").innerHTML = person.firstName;
```

- An array can have different types of variables.

  I.e Array can have arrays

  Array can have functions

  Array can have objects

```
myArray[0] = Date.now;
myArray[1] = myFunction;
myArray[2] = myCars;
```

- In Javascript, arrays with **named indexes** are **not supported**.
- If we use **named indexes** in javascript, arrays will be **redefined as objects**.

JavaScript Array Methods

Method	Description
concat()	Joins two or more arrays and returns a copy of the joined arrays
copyWithin()	Copies array elements within the array, to and from specified positions
entries()	Returns a key/value pair Array Iteration Object
every()	Checks if every element in an array pass a test
fill()	Fill the elements in an array with a static value
filter()	Creates a new array with every element in an array that pass a test
find()	Returns the value of the first element in an array that pass a test
findIndex()	Returns the index of the first element in an array that pass a test
forEach()	Calls a function for each array element
from()	Creates an array from an object
includes()	Check if an array contains the specified element
indexOf()	Search the array for an element and returns its position
isArray()	Checks whether an object is an array
join()	Joins all elements of an array into a string
keys()	Returns a Array Iteration Object, containing the keys of the original array
lastIndexOf()	Search the array for an element, starting at the end, and returns its position

Method	Description
map()	Creates a new array with the result of calling a function for each array element
pop()	Removes the last element of an array, and returns that element
push()	Adds new elements to the end of an array, and returns the new length
reduce()	Reduce the values of an array to a single value (going left-to-right)
reduceRight()	Reduce the values of an array to a single value (going right-to-left)
reverse()	Reverses the order of the elements in an array
shift()	Removes the first element of an array, and returns that element
slice()	Selects a part of an array, and returns the new array
some()	Checks if any of the elements in an array pass a test
sort()	Sorts the elements of an array
splice()	Adds/Removes elements from an array
toString()	Converts an array to a string, and returns the result
unshift()	Adds new elements to the beginning of an array, and returns the new length
valueOf()	Returns the primitive value of an array

JavaScript Array Properties

Property	Description
constructor	Returns the function that created the Array object's prototype
length	Sets or returns the number of elements in an array
prototype	Allows you to add properties and methods to an Array object