JavaScript Arrays

An array is an object that can store multiple values at once. For example,
var words = ['hello', 'world', 'welcome'];
Here, words is an array. The array is storing 3 values.

Create an Array

1. Using an array literal

The easiest way to create an array is by using an array literal `[]`. For example,

```
const array1 = ["eat", "sleep"];
```

2. Using the new keyword

You can also create an array using JavaScript's `new` keyword.

```
const array2 = new Array("eat", "sleep");
```

In both of the above examples, we have created an array having two elements.

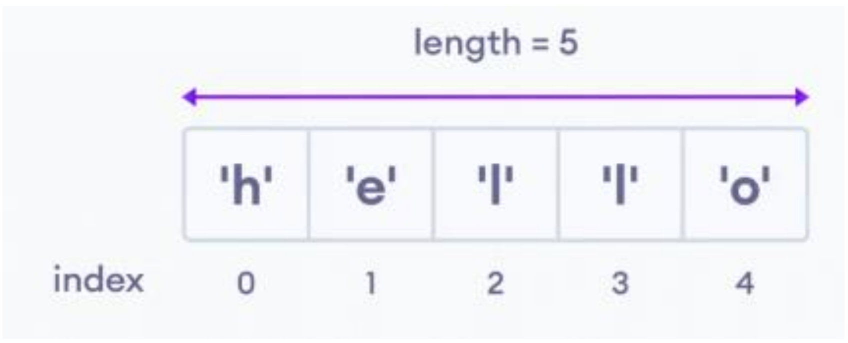
Access Elements of an Array

You can access elements of an array using indices **(0, 1, 2 …)**. For example,

```
const myArray = ['h', 'e', 'l', 'l', 'o'];

// first element
console.log(myArray[0]);  // "h"

// second element
console.log(myArray[1]); // "e"
```



Array Methods

In JavaScript, there are various array methods available that makes it easier to perform useful calculations.
Some of the commonly used JavaScript array methods are:

Method	Description
concat()	joins two or more arrays and returns a result
indexOf()	searches an element of an array and returns its position
find()	returns the first value of an array element that passes a test
findIndex()	returns the first index of an array element that passes a test
forEach()	calls a function for each element
includes()	checks if an array contains a specified element
push()	aads a new element to the end of an array and returns the new length of an array
unshift()	adds a new element to the beginning of an array and returns the new length of an array
pop()	removes the last element of an array and returns the removed element
shift()	removes the first element of an array and returns the removed element
sort()	sorts the elements alphabetically in strings and in ascending order
slice()	selects the part of an array and returns the new array
splice()	removes or replaces existing elements and/or adds new elements

Example: JavaScript Array Methods

```
let dailyActivities = ['sleep', 'work', 'exercise']
let newRoutine = ['eat'];

// sorting elements in the alphabetical order
dailyActivities.sort();
console.log(dailyActivities); // ['exercise', 'sleep', 'work']

//finding the index position of string
const position = dailyActivities.indexOf('work');
console.log(position); // 2

// slicing the array elements
const newDailyActivities = dailyActivities.slice(1);
console.log(newDailyActivities); // [ 'sleep', 'work']

// concatenating two arrays
const routine = dailyActivities.concat(newRoutine);
console.log(routine); // ["exercise", "sleep", "work", "eat"]
```

In JavaScript, an array is an object. And, the indices of arrays are objects keys.
Since arrays are objects, the array elements are stored by reference. Hence, when an array value is copied, any change in the copied array will also reflect in the original array. For example,

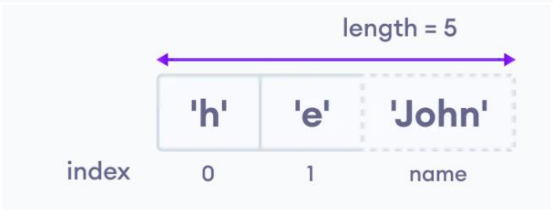
```
let arr = ['h', 'e'];
let arr1 = arr;
arr1.push('l');

console.log(arr); // ["h", "e", "l"]
console.log(arr1); // ["h", "e", "l"]
```

You can also store values by passing a named key in an array. For example,

```
let arr = ['h', 'e'];
arr.name = 'John';

console.log(arr); // ["h", "e"]
console.log(arr.name); // "John"
console.log(arr['name']); // "John"
```



JavaScript forEach()

In this tutorial, you will learn about JavaScript forEach() method with the help of examples.
The forEach() method calls a function and iterates over the elements of an array.
The forEach() method can also be used on Maps and Sets.

The syntax of the forEach() method is:

```
array.forEach(function(currentValue, index, arr))
```

- index (optional) - the index of the current element arr (optional) - the array of the current elements Here,
- function(currentValue, index, arr) - a function to be run for each element of an array
- currentValue - the value of an array

forEach with Arrays

The forEach() method is used to iterate over an array. For example,

```
let students = ['John', 'Sara', 'Jack'];

// using forEach
students.forEach(myFunction);

function myFunction(item) {

    console.log(item);
}
```

Updating the Array Elements

As we have seen in the above example, the forEach() method is used to iterate over an array, it is quite simple to update the array elements. For example,

```
let students = ['John', 'Sara', 'Jack'];

// using forEach
students.forEach(myFunction);

function myFunction(item, index, arr) {

    // adding strings to the array elements
    arr[index] = 'Hello ' + item;
}

console.log(students);
```

forEach with Arrow Function

You can use the arrow function with the forEach() method to write a program. For example,

```
// with arrow function and callback

const students = ['John', 'Sara', 'Jack'];

students.forEach(element => {
  console.log(element);
});
```

Run Cod

for loop to forEach()

Here is an example of how we can write a program with `for` loop and with `forEach()`.

Using for loop

```
const arrayItems = ['item1', 'item2', 'item3'];
const copyItems = [];

// using for loop
for (let i = 0; i < arrayItems.length; i++) {
  copyItems.push(arrayItems[i]);
}

console.log(copyItems);
```