Array

JavaScript provides a data type specifically for storing sequences of values. Array is a standard object of javascript. It is called an array and is written as below -

```
let listOfNumbers = [1, 2, 3, 4 , 5];
console.log(listOfNumbers[2]);
console.log(listOfNumbers[0]);
console.log(lisfOfNumbers[2-1]);
```

Finding items in array is easy -

```
const birds = ['Parrot', 'Falcon', 'Owl'];
console.log(birds.indexOf('Owl'));
console.log(birds.indexOf('Rabbit'));
```

Array has its own Length property. Like we can use string.length, we can know how many values are in an array by using length property.

```
let fruits = ["Apple", "Orange"]
fruits.length  //2
fruits[2] = "Mango"
fruits.length //3
```

There can be arrays within array -

```
let mix = [ [1, 2, 3], ['Ant', 'Cat', 'Dog'] ]
let nums = mix[0]
let animals = mix[1]
let x = mix[0][1]
let aungnet = mix[1][2]
```

Array Methods

To arrange array data, there are default methods in JavaScript. This four methods are main methods you need to memorize.

push() method

```
const cities = ['Manchester', 'Liverpool'];
cities.push('London');
console.log(cities);
cities.push('Bradford', 'Brighton');
```

We can know the added array by assigning it to a new variable.

```
const cities = ['Manchester', 'Liverpool'];
const newLength = cities.push('Bristol');
console.log(cities);     // [ "Manchester", "Liverpool", "Bristol" ]
console.log(newLength);  // 3
```

unshift() method

```
const cities = ['Manchester', 'Liverpool'];
cities.unshift('Edinburgh');
console.log(cities);     // [ "Edinburgh", "Manchester", "Liverpool" ]
```

pop() method

```
const cities = ['Manchester', 'Liverpool'];
cities.pop();
console.log(cities);
```

We can know the removed array by assigning it to a new variable.

```
const cities = ['Manchester', 'Liverpool'];
const removedCity = cities.pop();
console.log(removedCity);
```

shift() method

```
const cities = ['Manchester', 'Liverpool'];
cities.shift();
console.log(cities);     // [ "Liverpool" ]
```

If you know the index of an item, you can remove it from the array using splice()

```
const cities = ['Manchester', 'Liverpool', 'Edinburgh', 'Carlisle'];
const cutIndex = cities.indexOf('Liverpool');
if (cutIndex !== -1) {
  cities.splice(cutIndex, 1);
}
console.log(cities);
```

Let's take a real life example-

```
let todoList = [];
function remember(task) {
    todoList.push(task);
}

function getTask() {
    return todoList.shift();
}

function rememberUrgently(task) {
    todoList.unshift(task);
}
```

Accessing every items

When you want to access every item in the array. You can do for....of statement

```
const fruits = ['Mango', 'Banana', 'Mango'];

for (const fruit of fruits) {
    console.log(fruit)
}
```

There are other useful methods like above. But you might notice that the above methods change the original array make changes. This makes huge difference when you develop your website.

map() method

For map() method you need to give new function expression and they gave new array.

```
let nums = [1, 2, 3, 4, 5]

let result = nums.map(function(n) {
    return n + 1
})
```

You can write this as below-

```
function add(n) {
    return n + 1
}
const numbers = [1, 2, 3, 4, 5]
const added = numbers.map(add);
console.log(added);
```

filter() method

```
let nums = [1, 2, 3, 4, 5]
let result = nums.filter(n => n % 2)
```

Another example-

```
function isLong(city) {
    return city.length > 8;
}
const cities = ['London', 'Liverpool', 'Totnes', 'Edinburgh'];
const longer = cities.filter(isLong);
console.log(longer);  // [ "Liverpool", "Edinburgh" ]
```

Combining map() and filter() method

```
let nums = [1, 2, 3, 4, 5]
let result = nums.map(n => n + 2).filter(n => n % 2)
```

Reduce() method

```
const array1 = [1, 2, 3, 4];

// 0 + 1 + 2 + 3 + 4
const initialValue = 0;
const sumWithInitial = array1.reduce(
  (previousValue, currentValue) => previousValue + currentValue,
  initialValue
);

console.log(sumWithInitial);
// expected output: 10
```

In simple form -

```
let nums = [2, 3, 4, 5, 6]
let result =  nums.reduce((a,n) => a + n)
```

Converting between strings and arrays

Strings to array

```
const data = 'Manchester,London,Liverpool,Birmingham,Leeds,Carlisle';
const cities = data.split(',');
cities;
```

Array to strings

```
const dogNames = ['Rocket','Flash','Bella','Slugger'];
dogNames.toString(); // Rocket,Flash,Bella,Slugger
```

Array Spread & Destructuring

```
let nums = [1, 2, 3]
let alphas = ['a', 'b', 'c']
let result = [ nums, alphas ]
```

You can see the two arrays inside big array. If you want two arrays inside on it, you can do this like that.

```
let nums = [1, 2, 3]
let alphas = ['a', 'b', 'c']
let result = [ ...nums, ...alphas ]
```