Arrays

Array

- Arrays are generally described as "list-like objects"
- they are basically single objects that contain multiple values stored in a list
- neither the length nor types of its elements are fixed
- data can be stored at non-contiguous locations in the array
- cannot use strings as element indexes (as in an associative array) but must use integers

Creating and manipulating an Array

```
let shopping = ['bread', 'milk', 'noodles'];

console.log(shopping);

console.log(shopping[0]);

shopping[0] = 'cheese';
```

Creating an Array using **new**

```
let shopping = new Array('bread', 'milk', 'noodles');

console.log(shopping);

console.log(shopping[0]);

shopping[0] = 'cheese';
```

Create a new array from elements **(of)**

```
let arr = Array.of(1, "hello", {sub: "AT"})

console.log(arr[0]);  // 1

console.log(arr[1]);  //  "hello"

console.log(arr[2]);  // Object { sub: "AT" }

arr[33] = 4;

console.log(arr.length);   // 34
```

Check for an Array (**isArray**)

```
Array.isArray([1, 2, 3]);          // true

Array.isArray({num: 123});         // false

Array.isArray('foobar');           // false

Array.isArray(undefined);          // false
```

Looping array elements

```
let sequence = [1, 1, 2, 3, 5, 8, 13];

for (let x = 0; x < sequence.length; x++) {

  console.log(sequence[x]);

}
```

Looping array elements (Using **callback** function)

```
let sequence = [1, 1, 2, 3, 5, 8, 13];

sequence.forEach(disp);

function disp(item, index) {

    console.log(index, item)

}
```

Looping array elements (Using **callback** function)

```
let sequence = [1, 1, 2, 3, 5, 8, 13];

sequence.forEach(function(item, index) {

    console.log(index, item);

})
```

Looping array elements (Using **callback** function)

```
let sequence = [1, 1, 2, 3, 5, 8, 13];

sequence.forEach( (item, index) =>
console.log(index, item))
```

Looping array elements (Using **callback** function)

```
let sequence = [1, 1, 2, 3, 5, 8, 13];

sequence.forEach( item =>

console.log(item))
```

Merge two or more arrays (**concat**)

```
const array1 = ['a', 'b', 'c'];

const array2 = ['d', 'e', 'f'];

const array3 = array1.concat(array2);

console.log(array3);

//  ["a", "b", "c", "d", "e", "f"]
```

Add / remove item to / from **end** of an Array

```
let fruits = ['Apple', 'Banana']

let newLength = fruits.push('Orange')    // 3

// ["Apple", "Banana", "Orange"]

let last = fruits.pop()          // "Orange"

// ["Apple", "Banana"]
```

Add / remove item to / from **begining** of an Array

```
let fruits = ['Apple', 'Banana']

let first = fruits.shift()                  // "Apple"

//  ["Banana"]

let newLength = fruits.unshift('Berry')    //
2

// ["Berry", "Banana"]
```

Find the **index** of an item

```
let fruits = ['Apple', 'Banana']

let pos = fruits.indexOf('Banana')   // 1
```

Remove an item (**splice**)

```
let fruits = ["Strawberry", "Banana", "Mango"]

let pos = fruits.indexOf('Banana')

let removedItem = fruits.splice(pos, 1)   //
Array ["Banana"]

// ["Strawberry", "Mango"]
```

Note: Original array is **modified**

Insert and/or replace an item (**splice**)

```
const months = ['Jan', 'March', 'April', 'June'];

months.splice(1, 0, 'Feb');   // inserts at index 1

console.log(months);   // ["Jan", "Feb", "March",
"April", "June"]

months.splice(4, 1, 'May');  // replaces 1 element
at index 4

console.log(months);   // ["Jan", "Feb", "March",
"April", "May"]
```

Copy an array (**slice**)

```
let fruits = ["Strawberry", "Banana", "Mango"]

let shallowCopy = fruits.slice()    //

["Strawberry", "Banana", "Mango"]
```

Note: Original array is **not** modified

Copy sub-array elements (**slice**)

```
const animals = ['ant', 'bison', 'camel', 'duck',
'elephant'];

console.log(animals.slice(2));  // ["camel",
"duck", "elephant"]

console.log(animals.slice(2, 4)); // ["camel",
"duck"]

console.log(animals.slice(1, 5)); // ["bison",
"camel", "duck", "elephant"]
```

Converting Array Elements to String (**join**)

```
const elements = ['Fire', 'Air', 'Water'];

console.log(elements.join());

//  "Fire,Air,Water"

console.log(elements.join(''));

//  "FireAirWater"

console.log(elements.join('-'));

//  "Fire-Air-Water"
```

Converting String to an Array (**split**)

```
const str = 'This is a test string';

const words = str.split(' ');

console.log(words[3]); // "test"

const chars = str.split('');

console.log(chars[8]);  //  "a"

const strCopy = str.split();

console.log(strCopy);  // Array ["This is a test string"]
```

Check for an item belongs to an Array (**includes**)

```
const num = [1, 2, 3];

console.log(num.includes(2));     //  true

const pets = ['cat', 'dog', 'bat'];

console.log(pets.includes('cat'));    // true

console.log(pets.includes('at'));     // false
```

Check every items pass a criteria (**every**)

```
const arr = [1, 30, 39, 29, 10, 13];

console.log(arr.every(x => x < 40));

// true

console.log(arr.every(x => x < 30));

// false
```

Create new array which pass a criteria (**filter**)

```
const arr = [1, 30, 39, 29, 10, 13];

console.log(arr.filter(x => x < 30));

// [1, 29, 10, 13]

console.log(arr.filter(x => x < 20));

// [1, 10, 13]
```

Create new array (**map**)

The map() method creates a new array populated with the results of calling a provided function on every element in the calling array:

```
const array1 = [1, 4, 9, 16];

const array2 = array1.map(x => x * 2);

console.log(array2);// [2, 8, 18, 32]
```

Find the first matching element (**find**)

```
const array1 = [1, 4, 9, 16];

const found = array1.find(x => x > 5);

console.log(found);  // 9
```

Find the index of first matching element (**findIndex**)

```
const array1 = [1, 4, 9, 16];

const foundIndex = array1.findIndex(x =>
x > 5);

console.log(foundIndex);     // 2
```

Apply a function to each element (**reduce**)

Apply a reducer function to each element to produce a single value:

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

const reducer = (accumulator, curValue) => accumulator + curValue;

console.log(array1.reduce(reducer));  // 1 + 2 + 3 + 4 = 10

console.log(array1.reduce(reducer, 5)); // 5 + 1 + 2 + 3 + 4 = 15

// 5 is the initial value of an accumulator
```

Reversing array items in place (**reverse**)

```
const array1 = ['one', 'two', 'three'];

const reversed = array1.reverse();

console.log(reversed);          // ["three",
"two", "one"]

console.log(array1);    // ["three", "two",
"one"]
```

Sorting array items in place (**sort**)

- converts the elements into strings, then comparing their sequences of UTF-16 code units values:

```
const months = ['March', 'Jan', 'Feb', 'Dec'];

months.sort();

console.log(months);   // ["Dec", "Feb", "Jan", "March"]
```

```
const array1 = [1, 30, 4, 21, 1000];

array1.sort();

console.log(array1);        // [1, 1000, 21, 30, 4]
```

Sorting numbers (**sort**)

```
var numbers = [4, 2, 5, 1, 3];

numbers.sort(function(a, b) {
    return a - b;
});

console.log(numbers); // [1, 2, 3, 4, 5]
```

References

- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array