Arrays

Class	js
Created	@Jan 16, 2021 2:42 PM
Materials	
Reviewed	☐
Type	

Arrays are objects.

It stores lists of items. Items can be any type.

Each item can be accessed using its index/position in array.

SPARSE ARRAYS

Arrays can be missing elements.

arr[2] = 5; ⇒⇒⇒⇒⇒. 2 is the index number be careful.

arr.length =3;

arr[0] = undefined;

arr =[, , 5];

ARRAY CONSTRUCTUR - LITERAL ARRAYS

Constructors:

var arr = new Array (5) ⇒⇒⇒⇒ If you pass a one parameter && that parameter is a number ⇒⇒ Then that parameter will be deemed as the length of the array.

var arr = new Array(1,2,3, '12') ⇒ Otherwise they will be the array's elements.

Literal Array:

var arr = []

ARRAY METHODS

Add:

arr.push()

arr.unshift()

Delete:

arr.pop()

arr.shift()

Both shift and pop return undefined if the array is empty.

To remove items from anywhere in an array

arr.splice(a, b, c);

a = index number

b=how many elements are deleted after the given index number(including the index number-element)

c- this one is added to the arr starting at the index specified = a.

```
var arr = [0, 1, 2, 3];

var removed = arr.splice(1, 2, 'two items missing');

removed[0]; // 1
removed[1]; // 2

arr[0]; // 0
arr[1]; // 'two items missing'
arr[2]; // 3
```

arr.slice(a,b);

a=index number

b=index number

This one is takes the element until the second parameter represent the index number and gives a NEW ARRAY = CLONE.

The original array remains SAME.

arr == arr.slice(a,b) /// FALSE

```
var arr = [0, 1, 2, 3];

var sliced = arr.slice(1, 3);

sliced; // [1, 2]
```

join

Join converts an array into a string.

concat

returns a new array consisting of the combination of the passed in arrays and/or non-array values

reverse

reverses the array

sort

sorts the array. By default, it sorts alphabetically.

```
const months = ['March', 'Jan', 'Feb', 'Dec'];
months.sort();
console.log(months);
// expected output: Array ["Dec", "Feb", "Jan", "March"]

const array1 = [1, 30, 4, 21, 100000];
array1.sort();
console.log(array1);
// expected output: Array [1, 100000, 21, 30, 4]
```

filter: i.e word.length function

returns a new array consisting of only those elements that pass a test

map: i.e = multiply every el. by 2

returns a new array consisting of elements returned by the callback that is passed in

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

// pass a function to map
const map1 = array1.map(x => x * 2);

console.log(map1);
// expected output: Array [2, 8, 18, 32]
```

some : a single elements in the arrays passes a test

It returns a Boolean value.

```
const array = [1, 2, 3, 4, 5];

// checks whether an element is even
const even = (element) => element % 2 === 0;

console.log(array.some(even));
// expected output: true
```

every - if every item in the array passes the test

It returns a Boolean value.

```
const isBelowThreshold = (currentValue) => currentValue < 40;

const array1 = [1, 30, 39, 29, 10, 13];

console.log(array1.every(isBelowThreshold));
// expected output: true
```

indexOf

returns the index of an item in an array.

If the item appears in the array more than once then only the first index is returned.

If the item is not in the array then -1 is returned.

Note that === is used to test if the specified item is in the array.

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

console.log(beasts.indexOf('bison'));
// expected output: 1

// start from index 2
console.log(beasts.indexOf('bison', 2));
// expected output: 4

console.log(beasts.indexOf('giraffe'));
// expected output: -1
```

lastIndexOf

returns the index of the **last occuring match** in the array instead of the first

forEach - an alternative to a for loop

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

array1.forEach(element => console.log(element));

// expected output: "a"
// expected output: "b"
// expected output: "c"
```