Arrays



Overview

```
- what's an array?
- typeof []
- Array.isArray
- bracket access, bracket assignment
- .length property
- basic methods
```



What is an array?

```
/* An array is a list-like data structure in JavaScript */
let numbers = [1, 2, 3];
let names = ['George', 'John', 'Thomas'];
let aVariable = 'a value';
let mixedBag = [30, true, 'apples', null, aVariable];
/* the values inside of an array are called elements */
```



What is the typeof an array?

```
let names = ['George', 'John', 'Thomas'];

console.log(typeof names);

console.log(typeof names);
```



Array.isArray

```
let names = ['George', 'John', 'Thomas'];
console.log(Array.isArray(names));
console.log(Array.isArray('i am not an array'));
```

Bracket access

```
/* Access elements in an array the same way you'd access a character in a
 string: using brackets and the index number corresponding to the
 position of the element inside the array */
let names = ['George', 'John', 'Thomas'];
console.log(names[0]);
console.log(names[1]);
console.log(names[2]);
console.log(names[3]);
```



Bracket assignment

```
/* Use brackets and the assignment operator to assign new values to index
 positions in an array */
let names = ['George', 'John', 'Thomas'];
names[0] = 'Washington';
names[1] = 'Adams';
names[2] = 'Jefferson';
console.log(names);
```



.length property

```
/* Arrays, like strings, have a length property */
let names = ['George', 'John', 'Thomas'];
console.log(names.length)

console.log(names.length)

10
11
12
```





.push method

```
/* .push takes one or more elements and adds them to the end of the array.
  .push returns the new length of the array. */
let names = ['George', 'John', 'Thomas'];
let new length = names.push('James');
console.log(names);
console.log(newLength);
```



.pop method

```
/* .pop removes one element from the end of the array. it returns the removed element */

let names = ['George', 'John', 'Thomas'];

let jefferson = names.pop();

console.log(names);
console.log(jefferson)
```





.shift method

```
/* .shift works like .pop, but it removes the first element instead */
/* it's called shift because the indices for every element in the array
  are shifted over by one */
let names = ['George', 'John', 'Thomas'];
let washington = names.shift();
console.log(names);
console.log(washington);
```

.unshift method

```
/* .unshift adds one or more elements to the front of the array */
let names = ['George', 'John', 'Thomas'];
let newLength = names.unshift('King George III');
console.log(names);
console.log(newLength);
```



.indexOf method

```
/* .indexOf is also an array method, and works the same way as the string method of the same name */

let names = ['George', 'John', 'Thomas'];

console.log(names.indexOf('George'));

console.log(names.indexOf('Alexander'));
```



[John] [George, John, Thomas]

.slice method

```
/* .slice is also an array method, and works the same way as the string
method of the same name. */

let names = ['George', 'John', 'Thomas'];

let oneTermPresidents = names.slice(1, 2);

console.log(oneTermPresidents);
console.log(names);
```





.slice method

```
/* .slice is also an array method, and works the same way as the string
  method of the same name. */
let names = ['George', 'John', 'Thomas'];
let namesCopy = names.slice();
namesCopy[2] = 'Paul';
namesCopy.push('Ringo');
console.log(names);
console.log(namesCopy);
```





includes method

```
/* .includes takes a value, and returns true if the value is an element in
 the array */
let names = ['George', 'John', 'Thomas'];
console.log(names.includes('George'));
console.log(names.includes('Alexander'));
```





.reverse method

```
/* .reverse mutates (changes) the original array, reversing the order of
  its elements */
let names = ['George', 'John', 'Thomas'];
names.reverse();
console.log(names);
```



Recap

```
- what's an array?
- typeof []
- Array.isArray
- bracket access, bracket assignment
- .length property
- basic methods
```