JavaScript Array Methods with Definitions and Examples

push()

Definition: Adds one or more elements to the end of an array.

Example:

```
let fruits = ["apple", "banana"];
fruits.push("mango");
console.log(fruits); // ["apple", "banana", "mango"]
```

pop()

Definition: Removes the last element from an array and returns it.

Example:

```
let fruits = ["apple", "banana", "mango"];
fruits.pop();
console.log(fruits); // ["apple", "banana"]
```

unshift()

Definition: Adds one or more elements to the beginning of an array.

Example:

```
let fruits = ["banana", "mango"];
fruits.unshift("apple");
console.log(fruits); // ["apple", "banana", "mango"]
```

shift()

Definition: Removes the first element from an array and returns it.

Example:

```
let fruits = ["apple", "banana"];
fruits.shift();
console.log(fruits); // ["banana"]
```

concat()

Definition: Combines two or more arrays and returns a new array.

Example:

```
let a = [1, 2];
let b = [3, 4];
let result = a.concat(b);
console.log(result); // [1, 2, 3, 4]
```

join()
Definition: Joins all elements of an array into a string.

Example:
```
let fruits = ["apple", "banana"];
console.log(fruits.join(" - ")); // "apple - banana"
```

slice()
Definition: Returns a shallow copy of a portion of an array.

Example:
```
let nums = [1, 2, 3, 4, 5];
console.log(nums.slice(1, 4)); // [2, 3, 4]
```

splice()
Definition: Adds or removes elements from an array.

Example:
```
let fruits = ["apple", "banana", "orange"];
fruits.splice(1, 1, "grapes");
console.log(fruits); // ["apple", "grapes", "orange"]
```

indexOf()
Definition: Returns the first index of the specified element.

Example:
```
let items = ["a", "b", "c"];
console.log(items.indexOf("b")); // 1
```

lastIndexOf()
Definition: Returns the last index of the specified element.

Example:
```
let items = ["a", "b", "c", "b"];
console.log(items.lastIndexOf("b")); // 3
```

includes()
Definition: Checks if an array contains a certain element.

Example:
```
let fruits = ["apple", "banana"];
console.log(fruits.includes("banana")); // true
```

find()
Definition: Returns the first element that satisfies a condition.

Example:
```
let numbers = [5, 12, 8, 130];
```

```
let result = numbers.find(n => n > 10);
console.log(result); // 12
```

filter()

Definition: Returns all elements that match a condition.

Example:

```
let nums = [1, 2, 3, 4, 5];
let even = nums.filter(n => n % 2 === 0);
console.log(even); // [2, 4]
```

map()

Definition: Creates a new array with the result of a function on every element.

Example:

```
let nums = [1, 2, 3];
let squares = nums.map(n => n * n);
console.log(squares); // [1, 4, 9]
```

forEach()

Definition: Executes a function for each array element.

Example:

```
let names = ["John", "Jane", "Bob"];
names.forEach(name => console.log("Hello " + name));
```

reduce()

Definition: Reduces the array to a single value.

Example:

```
let nums = [1, 2, 3, 4];
let sum = nums.reduce((total, n) => total + n, 0);
console.log(sum); // 10
```

every()

Definition: Returns true if all elements satisfy the condition.

Example:

```
let ages = [22, 25, 30];
console.log(ages.every(age => age >= 18)); // true
```

some()

Definition: Returns true if at least one element satisfies the condition.

Example:

```
let marks = [40, 50, 35];
console.log(marks.some(m => m < 40)); // true
```

sort()
Definition: Sorts the elements of an array.

Example:
```
let nums = [4, 2, 1, 3];
nums.sort();
console.log(nums); // [1, 2, 3, 4]
```

reverse()
Definition: Reverses the order of elements.

Example:
```
let letters = ["a", "b", "c"];
letters.reverse();
console.log(letters); // ["c", "b", "a"]
```

toString()
Definition: Converts array to a string.

Example:
```
let colors = ["red", "green"];
console.log(colors.toString()); // "red,green"
```

Array.isArray()
Definition: Checks if a value is an array.

Example:
```
console.log(Array.isArray([1, 2, 3])); // true
console.log(Array.isArray("hello")); // false
```

length
Definition: Returns the total number of elements in the array.

Example:
```
let items = [10, 20, 30];
console.log(items.length); // 3
```