In JavaScript, arrays come with a variety of **built-in methods** that help you manipulate and interact with data efficiently. Here's a categorized list of **commonly used array methods**:

**1.Adding/Removing Elements**

| **Method** | **Description** |
| --- | --- |
| push() | Adds one or more elements to the **end** of the array. |
| pop() | Removes the **last** element from the array. |
| unshift() | Adds one or more elements to the **beginning** of the array. |
| shift() | Removes the **first** element from the array. |
| splice(start, deleteCount, item1, item2, ...) | Adds or removes elements at a specific index. |
| slice(start, end) | Returns a shallow copy of a portion of an array. |

**push()**

let arr = [1, 2];

arr.push(3);

console.log(arr); // [1, 2, 3]

**pop()**

let arr = [1, 2, 3];

arr.pop();

console.log(arr); // [1, 2]

**unshift()**

let arr = [2, 3];

arr.unshift(1);

console.log(arr); // [1, 2, 3]

**shift()**

let arr = [1, 2, 3];

arr.shift();

console.log(arr); // [2, 3]

**splice()**

let arr = [1, 2, 3, 4];

arr.splice(2, 1, 10);

console.log(arr); // [1, 2, 10, 4]

**slice()**

let arr = [1, 2, 3, 4];

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

console.log(sliced); // [2, 3]

**2.Searching & Filtering**

| **Method** | **Description** |
| --- | --- |
| includes(value) | Checks if an array includes a certain value. |
| indexOf(value) | Returns the first index of the value (or -1 if not found). |
| lastIndexOf(value) | Returns the last index of the value. |
| find(callback) | Returns the **first** element that satisfies the condition. |
| findIndex(callback) | Returns the index of the first element satisfying the condition. |
| filter(callback) | Returns a **new array** with elements that pass the test. |

**includes()**

let arr = [1, 2, 3];

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

**indexOf()**

let arr = ['a', 'b', 'c'];

console.log(arr.indexOf('b')); // 1

**lastIndexOf()**

let arr = ['a', 'b', 'a'];

console.log(arr.lastIndexOf('a')); // 2

**find()**

let arr = [5, 12, 8];

let result = arr.find(n => n > 10);

console.log(result); // 12

**findIndex()**

let arr = [5, 12, 8];

let index = arr.findIndex(n => n > 10);

console.log(index); // 1

**filter()**

let arr = [1, 2, 3, 4];

let evens = arr.filter(n => n % 2 === 0);

console.log(evens); // [2, 4]

**3.Looping/Iteration**

| **Method** | **Description** |
| --- | --- |
| forEach(callback) | Executes a function for **each element** (no return). |
| map(callback) | Returns a **new array** with the results of applying a function to each element. |
| reduce(callback, initialValue) | Reduces the array to a single value. |
| some(callback) | Returns true if **at least one** element passes the test. |
| every(callback) | Returns true if **all** elements pass the test. |

**forEach()**

let arr = [1, 2, 3];

arr.forEach(n => console.log(n));

// 1

// 2

// 3

**map()**

let arr = [1, 2, 3];

let doubled = arr.map(n => n \* 2);

console.log(doubled); // [2, 4, 6]

**reduce()**

let arr = [1, 2, 3, 4];

let sum = arr.reduce((acc, curr) => acc + curr, 0);

console.log(sum); // 10

**some()**

let arr = [1, 2, 3];

console.log(arr.some(n => n > 2)); // true

**every()**

let arr = [2, 4, 6];

console.log(arr.every(n => n % 2 === 0)); // true

**4.Modifying/Combining**

| **Method** | **Description** |
| --- | --- |
| concat(array) | Combines two or more arrays into one. |
| join(separator) | Joins all elements into a string, with an optional separator. |
| reverse() | Reverses the order of the array in place. |
| sort() | Sorts the array alphabetically (or with a compare function). |
| flat(depth) | Flattens nested arrays up to specified depth. |
| fill(value, start, end) | Fills the array with a static value. |
| copyWithin(target, start, end) | Copies part of the array to another location in the same array. |

**concat()**

let a = [1, 2];

let b = [3, 4];

let result = a.concat(b);

console.log(result); // [1, 2, 3, 4]

**join()**

let arr = ['a', 'b', 'c'];

console.log(arr.join('-')); // 'a-b-c'

**reverse()**

let arr = [1, 2, 3];

arr.reverse();

console.log(arr); // [3, 2, 1]

**sort()**

let arr = [3, 1, 2];

arr.sort();

console.log(arr); // [1, 2, 3]

**flat()**

let arr = [1, [2, [3]]];

console.log(arr.flat(2)); // [1, 2, 3]

**fill()**

let arr = [1, 2, 3];

arr.fill(0);

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

**copyWithin()**

let arr = [1, 2, 3, 4, 5];

arr.copyWithin(1, 3);

console.log(arr); // [1, 4, 5, 4, 5]