JS Array

Introduction to Arrays

- **Definition**: An array is a list-like object used to store multiple values in a single variable.
- **Creating Arrays**:

```
let fruits = ['apple', 'banana', 'orange'];
let numbers = new Array(1, 2, 3);
```

2. Accessing Array Elements

- **Access by Index**:

```
let firstFruit = fruits[0]; // 'apple'
let secondNumber = numbers[1]; //
```

3. Array Methods

Adding and Removing Elements

- `push()`: Adds one or more elements to the end of an array.

```
fruits.push('grape'); // ['apple', 'banana', 'orange', 'grape']
```

- `pop()`: Removes the last element from an array.

```
let lastFruit = fruits.pop(); // 'grape'
```

- `unshift()`: Adds one or more elements to the beginning of an array.

```
fruits.unshift('kiwi'); // ['kiwi', 'apple', 'banana', 'orange']
```

```
let firstFruit = fruits.shift(); // 'kiwi'
```

- `shift()`: Removes the first element from an array.

```
fruits.splice(1, 1, 'mango'); // ['apple', 'mango', 'orange']
// Parameters: start index, number of elements to remove, element(s) to add
```

- `splice()`: Adds or removes elements from any position in an array.

Array Iteration

- `forEach()`: Executes a provided function once for each array element.

```
fruits.forEach(fruit => console.log(fruit));
```

- `map()`: Creates a new array with the results of calling a provided function on every element.

```
let upperFruits = fruits.map(fruit => fruit.toUpperCase()); // ['APPLE', 'BANANA', 'ORANGE']
```

- `filter()`: Creates a new array with all elements that pass the test implemented by the provided function.

```
let longFruits = fruits.filter(fruit => fruit.length > 5); // ['banana', 'orange']
```

- `reduce()`: Executes a reducer function on each element of the array, resulting in a single output value.

```
let totalLength = fruits.reduce((acc, fruit) => acc + fruit.length, 0); // 15
```

- `find()`: Returns the first element that satisfies the provided testing function.

```
let foundFruit = fruits.find(fruit => fruit.startsWith('b')); // 'banana'
```

- `some()`: Tests whether at least one element in the array passes the provided function.

```
let hasLongFruit = fruits.some(fruit => fruit.length > 5); // true
```

- `every()`: Tests whether all elements in the array pass the provided function.

```
let allShortFruits = fruits.every(fruit => fruit.length <= 6); // false
```

Array Sorting and Reversing

- `sort()`: Sorts the elements of an array in place and returns the sorted array.

```
fruits.sort(); // ['apple', 'banana', 'orange']
```

- `reverse()`: Reverses the elements of an array in place.

```
fruits.reverse(); // ['orange', 'banana', 'apple']
```

Finding and Joining Elements

- `indexOf()`: Returns the first index at which a given element can be found.

```
let index = fruits.indexOf('banana'); // 1
```

- `includes()`: Determines whether an array contains a certain element.

```
let hasApple = fruits.includes('apple'); // true
```

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

```
let fruitString = fruits.join(', '); // 'apple, banana, orange'
```

Array Copying

- `slice()`: Returns a shallow copy of a portion of an array into a new array object.

```
let slicedFruits = fruits.slice(1, 3); // ['banana', 'orange']
```

- `concat()`: Merges two or more arrays.

```
let moreFruits = fruits.concat(['grape', 'pear']); // ['apple', 'banana', 'orang
e', 'grape', 'pear'
```