Mastering JavaScript

Array Methods

Brief overview of what will be covered:

1. In this carousel, we'll explore the most commonly used JavaScript array methods.

2. Each method will be explained with clear examples to show how they can be applied in real-world scenarios.

3. By the end of this presentation, you'll have a solid understanding of how to use these methods to write more efficient and effective code.

Sayyed Siddique
sayyed-siddique-834245255

Swipe

1 Basic Array Methods

1. Push()

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

Example:

```
let arr = [1, 2, 3];
arr.push(4); // arr is now [1, 2, 3, 4]
```

Sayyed Siddique

sayyed-siddique-834245255

Basic Array Methods

2. Pop()

Description: Removes the last element from an array.

Example:

```
let arr = [1, 2, 3, 4];
arr.pop(); // arr is now [1, 2, 3]
```

Sayyed Siddique

sayyed-siddique-834245255

Basic Array Methods

3. Shift()

Description: Removes the first element from an array.

Example:

```
let arr = [1, 2, 3, 4];
arr.shift(); // arr is now [2, 3, 4]
```

Sayyed Siddique

sayyed-siddique-834245255

Basic Array Methods

4. Unshift()

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

Example:

```
let arr = [2, 3, 4];
arr.unshift(1); // arr is now [1, 2, 3, 4]
```

Sayyed Siddique

sayyed-siddique-834245255

2 Accessing and Iterating Methods

1. IndexOf()

Description: Returns the first index at which a given element can be found.

Example:

```
let arr = ['a', 'b', 'c', 'a'];
arr.indexOf('a'); // returns 0
```

Sayyed Siddique
sayyed-siddique-834245255

Accessing and Iterating Methods

2. lastIndexOf()

Description: Returns the last index at which a given element can be found.

Example:

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

arr.lastIndexOf('a'); // returns 3
```

Sayyed Siddique

sayyed-siddique-834245255

Accessing and Iterating Methods

3. ForEach()

Description: Executes a provided function once for each array element.

Example:

```
let arr = [1, 2, 3];
arr.forEach(element => console.log(element)); // prints 1 2 3
```

Sayyed Siddique
sayyed-siddique-834245255

Accessing and Iterating Methods

4. Find()

Description: Returns the first element that satisfies the provided testing function.

Example:

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

arr.find(element => element > 2); // returns 3
```

Sayyed Siddique

sayyed-siddique-834245255

Accessing and Iterating Methods

5. FindIndex()

Description: Returns the index of the first element that satisfies the testing function.

Example:

```
let arr = [1, 2, 3, 4];
arr.findIndex(element => element > 2); // returns 2
```

Sayyed Siddique

sayyed-siddique-834245255

3 Transforming Methods

1. map()

Description: Creates a new array with the results of calling a provided function on every element.

Example:

```
let arr = [1, 2, 3];
let doubled = arr.map(x => x * 2); // [2, 4, 6]
```

Sayyed Siddique
sayyed-siddique-834245255

Transforming Methods

2. Filter()

Description: Creates a new array with all elements that pass the test implemented by the provided function.

filter() method does not execute the function for empty elements.

Example:

```
let arr = [1, 2, 3, 4];
let filtered = arr.filter(x => x > 2); // [3, 4]
```

Sayyed Siddique
sayyed-siddique-834245255

Transforming Methods

2. Reduce()

Description: Executes a reducer function on each element of the array, **reduce()** method returns a single value

reduce() method does not change the original array.

Example:

```
let arr = [1, 2, 3, 4];
let sum = arr.reduce((acc, x) => acc + x, 0); // 10
```

Sayyed Siddique
sayyed-siddique-834245255

4 Sorting and Reversing Methods

1. Sort()

Description: Sorts the elements of an array in place and returns the sorted array.
sort() method sorts the elements as strings in alphabetical and ascending order.

sort() method overwrites the original array.

Example:

```
let arr = [3, 1, 4, 1, 2];
arr.sort(); // [1, 1, 2, 3, 4]
```

You can learn more about from internet

Sayyed Siddique
sayyed-siddique-834245255

Sorting and Reversing Methods

2. Reverse()

Description: Reverses the order of the elements of an array in place

reverse() method overwrites the original array.

Example:

```
let arr = ['a', 'b', 'c'];
let result = arr.reduceRight((acc, x) => acc + x, ''); // 'cba'
```

Sayyed Siddique
sayyed-siddique-834245255

5 Concatenation and Slicing Methods

1. Concat()

Description: Merges two or more arrays. concat() method returns a new array, containing the joined arrays.

concat() method does not change the existing arrays.

Example:

```
let arr1 = [1, 2];
let arr2 = [3, 4];
let merged = arr1.concat(arr2); // [1, 2, 3, 4]
```

Sayyed Siddique

sayyed-siddique-834245255

Concatenation and Slicing Methods

3. Splice()

Description: Changes the contents of an array by removing or replacing existing elements and/or adding new elements.

splice() method overwrites the original array.

Example:

```
let arr = [1, 2, 3, 4];
arr.splice(1, 2, 'a', 'b'); // arr is now [1, 'a', 'b', 4]
```

Sayyed Siddique
sayyed-siddique-834245255

6 Utility Methods

1. Includes()

Description: Determines whether an array includes a certain element. returns **true** if an array contains a specified value returns **false** if the value is not found.

includes() method is case sensitive.

Example:

```
let arr = [1, 2, 3];
arr.includes(2); // true
```

Sayyed Siddique
sayyed-siddique-834245255

Utility Methods

2. Join()

Description: Joins all elements of an array into a string.

Example:

```
let arr = [1, 2, 3];
arr.join('-'); // '1-2-3'
```

Sayyed Siddique
sayyed-siddique-834245255

Utility Methods

3. FlatMap()

Description: maps all array elements and creates a new flat array.

flatMap() does not change the original array.

Example:

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

arr.flatMap(x => [x * 2]); // [2, 4, 6]
```

Sayyed Siddique

sayyed-siddique-834245255

7 Array Methods for Searching and Checking

1. Every()

Description: Tests whether all elements in the array pass the test implemented by the provided function.

every() method does not change the original array

Example:

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

arr.every(x => x > 0); // true
```

Sayyed Siddique
sayyed-siddique-834245255

Array Methods for Searching and Checking

2. Some()

Description: Tests whether at least one element in the array passes the test implemented by the provided function.

some() method does not change the original array.

Example:

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

arr.some(x => x > 2); // true
```

Sayyed Siddique

sayyed-siddique-834245255

Array Methods for Searching and Checking

3. isArray()

Description: Determines whether the passed value is an array.

Example:

```
Array.isArray([1, 2, 3]); // true
Array.isArray('string'); // false
```

Sayyed Siddique

sayyed-siddique-834245255