Set in JS

About Set Data Structure

- The Set data structure was introduced in ECMAScript 2015 (ES6) and is supported in modern browsers and JavaScript environments.
- The Set object is a built-in data structure in JavaScript that allows you to store unique values of any type, whether they are primitive values or object references.
- It is an unordered collection of elements with no duplicates.

Creating new Set

- To create a new Set, you can use the `new` keyword with `Set()` constructor.
  For example:

```
const mySet = new Set();
```

- You can also initialize a Set with an array, by passing it as an argument to the constructor.
  For example:

```
const mySet = new Set([1, 2, 3]);
```

Key features of Set:

1. **Uniqueness:** A Set can only contain unique elements. If you try to add a duplicate value, it will be ignored.
2. **No Indexing:** Unlike arrays, Set elements are not accessed by an index. Instead, you can check if a specific value exists in the set using the `has()` method.

3. **Iteration**: You can easily iterate over the elements of a Set using the `for...of` loop or by converting it to an array using the `Array.from()` or `spread operator (...)`.
4. **Size:** The number of elements in a Set can be obtained using the `size` property.
5. **Adding and Removing Elements:** Elements can be added to a Set using the `add()` method and removed using the `delete()` method.
6. **Checking Element Existence:** You can check if an element exists in a Set using the `has()` method, which returns true if the element is found and false otherwise.
7. **Clearing the Set:** The `clear()` method removes all elements from a Set, leaving it empty.
8. **Equality Comparison:** Set objects are not considered equal to each other, even if they have the same elements. Each Set instance is unique.

Here's an example usage of the Set data structure in JavaScript:

```
const mySet = new Set();
mySet.add(1);
mySet.add(2);
mySet.add(3);
mySet.add(2); // Ignored, already exists

console.log(mySet.size); // Output: 3
console.log(mySet.has(2)); // Output: true

mySet.delete(3);
console.log(mySet.size); // Output: 2

mySet.clear();
console.log(mySet.size); // Output: 0
```

Benefits of the Set Data Structure

- The Set data structure is useful when storing unique values or performing operations like checking for duplicates or removing duplicates from an array.
- It provides an efficient and straightforward way to handle collections of unique elements in JavaScript.