

Lesson 03 Demo 07

Working with a HashMap

Objective: To demonstrate the HashMap functionality in JavaScript by building a key-value store, performing dynamic updates through add and delete operations, and managing map data using clear and display methods for efficient data handling

Tools required: Visual Studio Code and Node.js

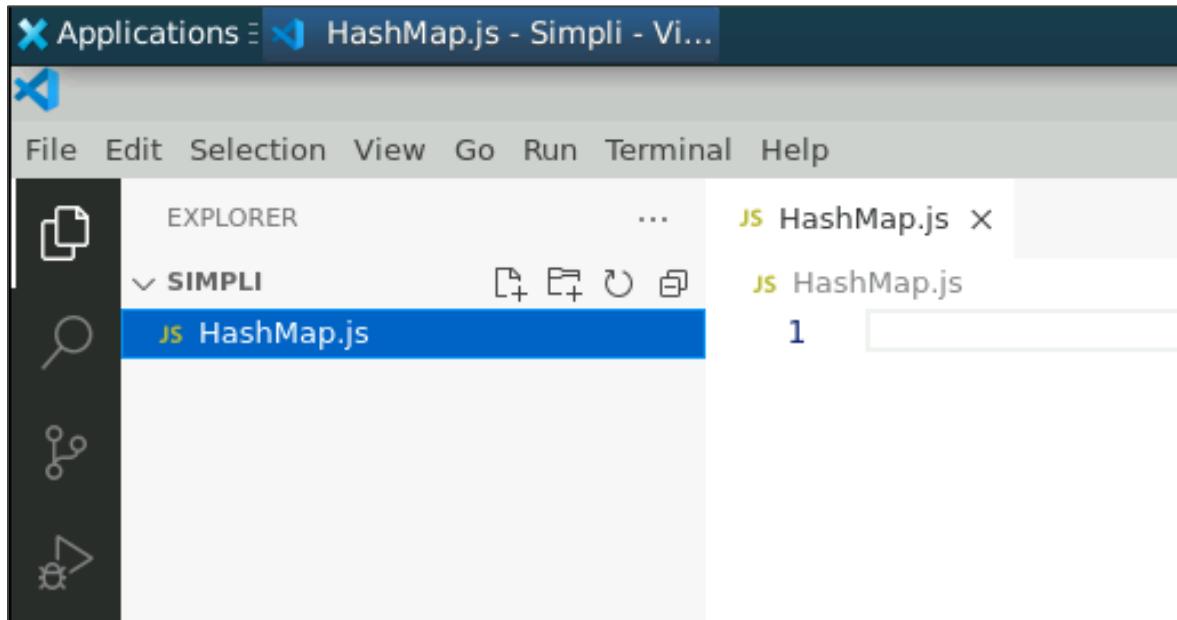
Prerequisites: A basic understanding of data structures and JavaScript

Steps to be followed:

1. Create a JavaScript file and execute it

Step 1: Create a JavaScript file and execute it

1.1 Open the Visual Studio Code editor and create a JavaScript file named **HashMap.js**



1.2 Add the following code to the file:

```
// Creating a HashMap
let hashMap = new Map();

// Adding key-value pairs to the HashMap
hashMap.set('key1', 'value1');
hashMap.set('key2', 'value2');
hashMap.set('key3', 'value3');

// Displaying the values in the HashMap
console.log('HashMap after adding elements:');
for (let [key, value] of hashMap) {
    console.log(`Key: ${key}, Value: ${value}`);
}

// Removing a key from the HashMap
hashMap.delete('key2');
console.log('HashMap after deleting key2:');

// Displaying the values in the HashMap after deletion
for (let [key, value] of hashMap) {
    console.log(`Key: ${key}, Value: ${value}`);
}

// Clearing all elements from the HashMap
hashMap.clear();
console.log('HashMap after clearing all elements:');

// Displaying the values in the HashMap after clearing
for (let [key, value] of hashMap) {
    console.log(`Key: ${key}, Value: ${value}`);
}
```

```
JS HashMap.js > ...
1 // Creating a HashMap
2 let hashMap = new Map();
3
4 // Adding key-value pairs to the HashMap
5 hashMap.set('key1', 'value1');
6 hashMap.set('key2', 'value2');
7 hashMap.set('key3', 'value3');
8
9 // Displaying the values in the HashMap
10 console.log('HashMap after adding elements:');
11 for (let [key, value] of hashMap) {
12     console.log(`Key: ${key}, Value: ${value}`);
13 }
14
15 // Removing a key from the HashMap
16 hashMap.delete('key2');
17 console.log('HashMap after deleting key2:');
18
19 // Displaying the values in the HashMap after deletion
20 for (let [key, value] of hashMap) {
21     console.log(`Key: ${key}, Value: ${value}`);
22 }
23
```

```
24 // Clearing all elements from the HashMap
25 hashMap.clear();
26 console.log('HashMap after clearing all elements:');
27
28 // Displaying the values in the HashMap after clearing
29 for (let [key, value] of hashMap) {
30     console.log(`Key: ${key}, Value: ${value}`);
31 }
32
```

1.3 Press **Ctrl + S** to save the file and execute it in the **TERMINAL** using the commands given below:

```
ls  
node HashMap.js
```

```
23  
24 // Clearing all elements from the HashMap  
  
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL  
  
priyanshurajsim@ip-172-31-39-132:~/Downloads/Simpli$ ls  
HashMap.js  
priyanshurajsim@ip-172-31-39-132:~/Downloads/Simpli$ node HashMap.js  
HashMap after adding elements:  
Key: key1, Value: value1  
Key: key2, Value: value2  
Key: key3, Value: value3  
HashMap after deleting key2:  
Key: key1, Value: value1  
Key: key3, Value: value3  
HashMap after clearing all elements:  
priyanshurajsim@ip-172-31-39-132:~/Downloads/Simpli$ █
```

By following these steps, you have implemented `HashMap` operations in JavaScript, including adding, deleting, clearing, and iterating elements. This example demonstrates declaring, initializing, accessing values, and checking key existence for efficient data handling.