Lesson 03 Demo 07

Working with HashMap

Objective: To demonstrate the usage of a HashMap in JavaScript, covering the creation of a HashMap, addition and deletion of key-value pairs, and the methods to clear and display its contents

Tools required: Visual Studio Code and Node.js

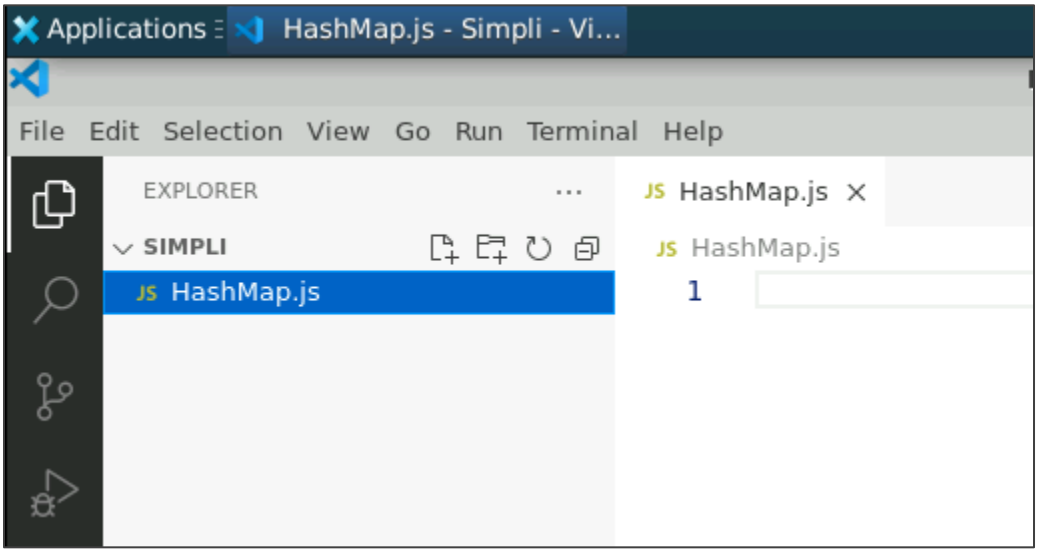
Prerequisites: Basic understanding of data structures and JavaScript

Steps to be followed:

1. Create and execute the JS file

Step 1: Create and execute the JS file

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



1.2 Write the code given below in the **HashMap.js** 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 Save the file and execute it in the terminal using the command given below:
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$
```

This example shows how to declare a HashMap, initialize it with key-value pairs, access specific values, and check if a key exists in the HashMap in JavaScript.

By following these steps, you have successfully implemented operations on a HashMap in JavaScript, including adding, deleting, and clearing elements, as well as iterating over its contents, enhancing your skills in data handling and manipulation in programming.