Lesson 04 Demo 01

Implementing String Methods

Objective: To implement the essential aspects of JavaScript strings, encompassing syntax variations, properties, and methods for a comprehensive understanding of string manipulation

Tools required: Visual Studio Code and Node.js

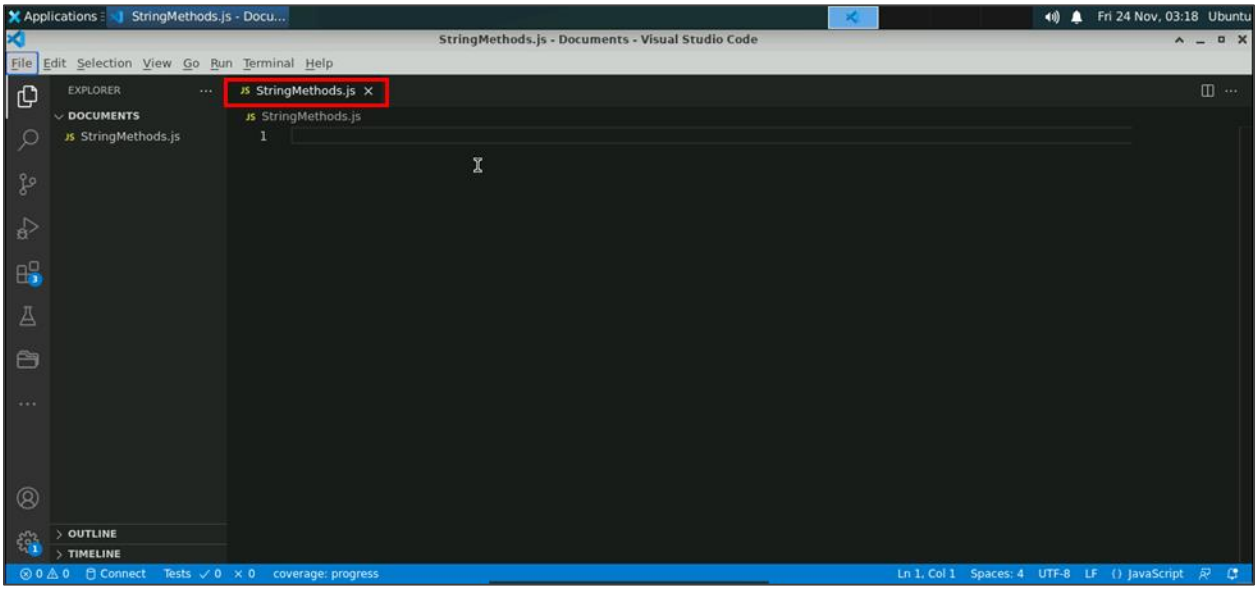
Prerequisites: A basic understanding of string properties and methods in JavaScript

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 **StringMethods.js**

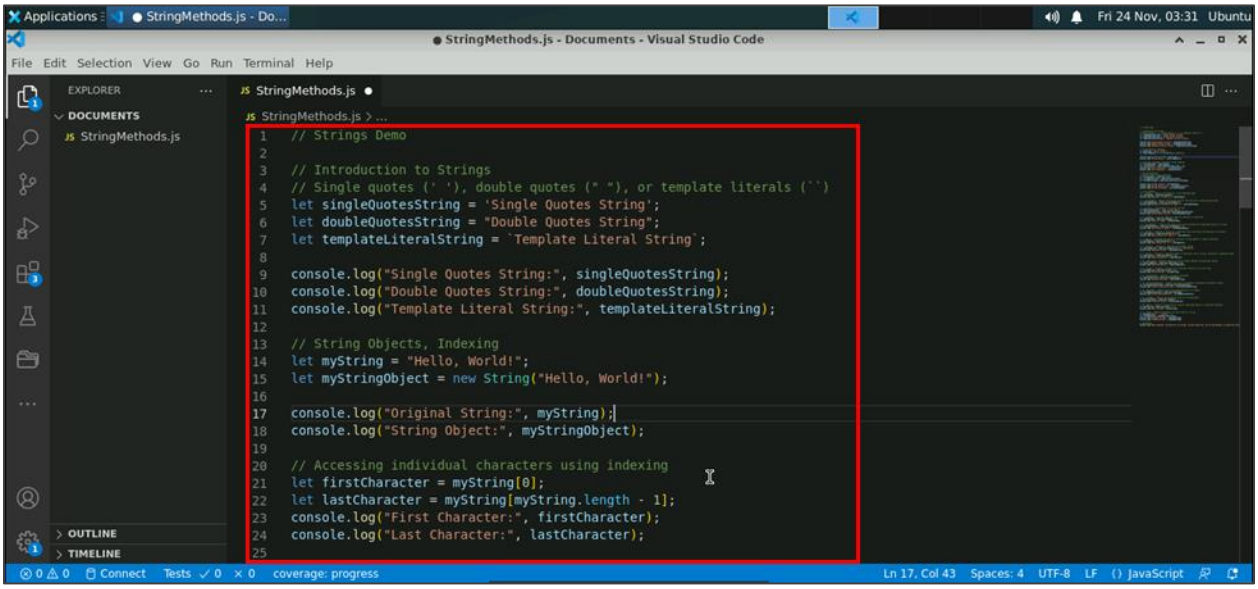


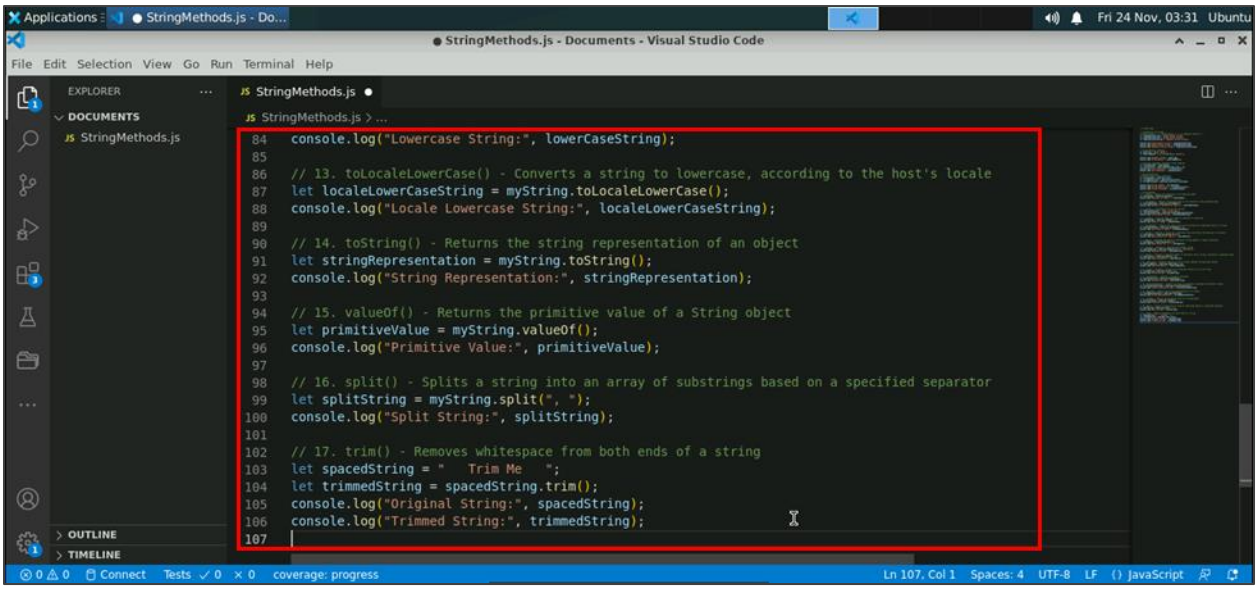
1.2 Enter the code below into the **StringMethods.js** file:

```
// Single quotes (' '), double quotes (" "), or template literals (``)
let singleQuotesString = 'Single Quotes String';
let doubleQuotesString = "Double Quotes String";
let templateLiteralString = `Template Literal String`;

console.log("Single Quotes String:", singleQuotesString);
console.log("Double Quotes String:", doubleQuotesString);
console.log("Template Literal String:", templateLiteralString);

// String Objects, Indexing
let myStringObject = new String("Hello, World!");

console.log("String Object:", myStringObject);

// Accessing individual characters using indexing
let firstCharacter = myStringLiteral[0];
let lastCharacter = myStringLiteral[myStringLiteral.length - 1];
console.log("First Character:", firstCharacter);
console.log("Last Character:", lastCharacter);

// String Properties
// Demonstrating string properties
let stringLength = myString.length;
let stringConstructor = myString.constructor;
let stringPrototype = myString.constructor.prototype;

console.log("String Length:", stringLength);
console.log("String Constructor:", stringConstructor);
console.log("String Prototype:", stringPrototype);

// String Methods
// 1. charAt() - Returns the character at the specified index
let charAtIndex = myString.charAt(7);
console.log("Character at Index 7:", charAtIndex);

// 2. charCodeAt() - Returns the Unicode value of the character at the specified index
let charCodeAtIndex = myString.charCodeAt(7);
console.log("Unicode of Character at Index 7:", charCodeAtIndex);

// 3. concat() - Concatenates two or more strings
let additionalString = " Have a great day!";
let concatenatedString = myString.concat(additionalString);
```

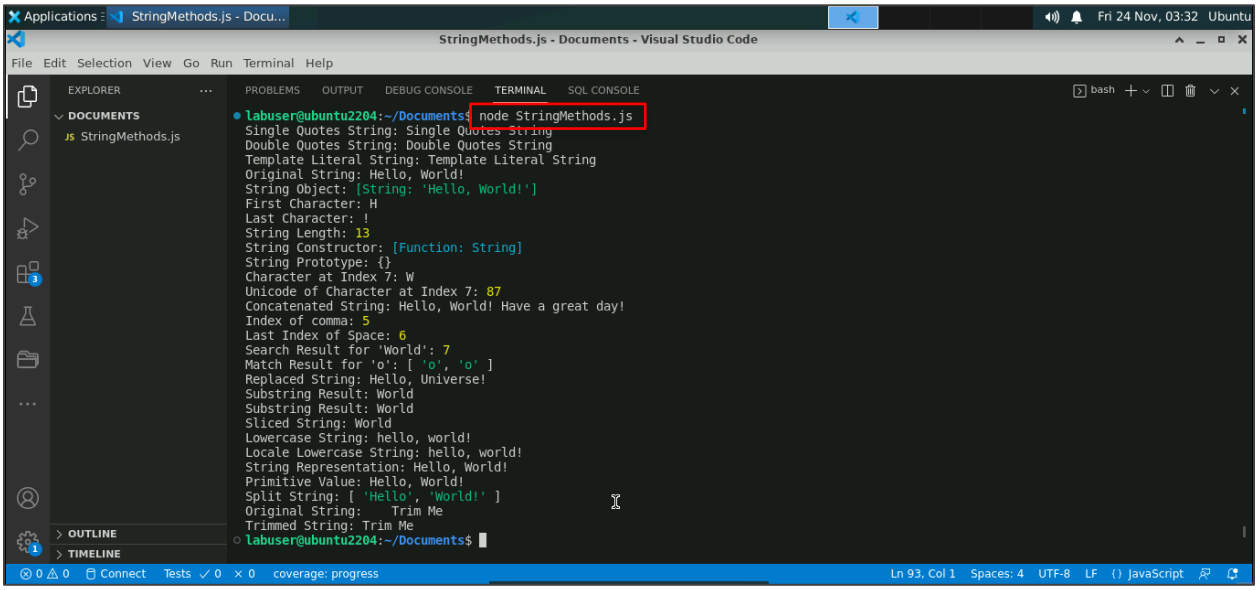
```
console.log("Concatenated String:", concatenatedString);

// 4. indexOf() - Find the index of a specific character or substring
let indexOfComma = myString.indexOf(",");
console.log("Index of comma:", indexOfComma);

// 5. lastIndexOf() - Returns the index of the last occurrence of a specified value in a string
let lastIndexOfSpace = myString.lastIndexOf(" ");
console.log("Last Index of Space:", lastIndexOfSpace);

// 6. search() - Searches a string for a specified value and returns the position of the match
let searchResult = myString.search("World");
console.log("Search Result for 'World':", searchResult);

// 7. match() - Retrieves the result of matching a string against a regular expression
let matchResult = myString.match(/o/g);
console.log("Match Result for 'o':", matchResult);

// 8. replace() - Replaces a substring with another string
let replacedString = myString.replace("World", "Universe");
console.log("Replaced String:", replacedString);

// 9. substr() - Extracts a specified number of characters from a string, starting at a specified index
let substrResult = myString.substr(7, 5);
console.log("Substring Result:", substrResult);

// 10. substring() - Extracts characters from a string, between two specified indices
let substringResult = myString.substring(7, 12);
console.log("Substring Result:", substringResult);

// 11. slice() - Extracts a section of a string and returns it as a new string
let slicedString = myString.slice(7, 12);
console.log("Sliced String:", slicedString);

// 12. toLowerCase() - Converts a string to lowercase
let lowerCaseString = myString.toLowerCase();
console.log("Lowercase String:", lowerCaseString);
```

```
// 13. toLocaleLowerCase() - Converts a string to lowercase, according to the host's locale
let localeLowerCaseString = myString.toLocaleLowerCase();
console.log("Locale Lowercase String:", localeLowerCaseString);

// 14. toString() - Returns the string representation of an object
let stringRepresentation = myString.toString();
console.log("String Representation:", stringRepresentation);

// 15. valueOf() - Returns the primitive value of a String object
let primitiveValue = myString.valueOf();
console.log("Primitive Value:", primitiveValue);

// 16. split() - Splits a string into an array of substrings based on a specified separator
let splitString = myString.split(", ");
console.log("Split String:", splitString);

// 17. trim() - Removes whitespace from both ends of a string
let spacedString = "   Trim Me   ";
let trimmedString = spacedString.trim();
console.log("Original String:", spacedString);
console.log("Trimmed String:", trimmedString);
```





1.3 Save the file and execute the command given below in the terminal to run it:
node StringMethods.js



The provided code demonstrates string creation with various quotes and explores properties like length. It uses string methods for manipulation, including indexing, searching, and case conversion in JavaScript.

By following the above steps, you have successfully demonstrated the usage of diverse string creation methods and properties, such as length, and various manipulation techniques for effective string handling in JavaScript.