SOC-Memory Game

Charishma Gowdu

June 17, 2024

1 Introduction

We had a good knowledge about the basics of Javascript in the last week's assignment. So this week we are going to make a report of some advanced topics of javascript.But yeah there will be some things which was related to 1st week's assaignment so that we'll have all the stuff handy.

2 Objects

An object literal is a list of name:value pairs inside curly braces .

2.1 Object Properties

2.1.1 Creation of Object:

- const person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
- const person = { firstName: "John",
  lastName: "Doe",

age: 50,

eyeColor: "blue" };

- const person = {};

  person.firstName = "John";

  person.lastName = "Doe";

  person.age = 50;

  person.eyeColor = "blue";

2.2 Accessing Javascript Properties

Example: let age = person.age; or

let age = person[age]; or

let age = person[x];

2.2.1 Adding New Properties

Example: person.nationality = "English";

2.2.2 Deleting Properties

Example: delete person.age (or)

delete person["age"];

2.3 Object Methods

const person = { firstName: "John",

lastName: "Doe",

id: 5566,

fullName: function() {

return this.firstName + " " + this.lastName;

}

};

So here, **this** refers to the person object.

this.firstName means the firstName property of person.

2.4 Accessing Object Methods

We access an object method with the syntax: objectName.methodName()

For example: name = person.fullName()

2.5 Advantage of JavaScript Methods

We can use **toUpperCase()** method to convert a text to uppercase

3 JavaScript Display Objects

- Object.values() creates an array from the property values
- Object.entries() makes it simple to use objects in loops
- Javascript objects can be converted to a string with JSON method **JSON.stringify()**

4 JavaScript Object Constructors

Sometimes we need to create many objects of the same type.

To create an object type we use an object constructor function. **For Example:**

For example there is a object type function called **Person** and so now we want to create many new Person objects ,for that we can use **new Person()**

5 Common HTML Events

onchange : An HTML element has been changed

onclick : The user clecks an HTML element

onmouseover: The user moves the mouse over an HTML element

onmouseout : The user moves the mouse away from an HTML element

onkeydown : The user pushes a keyboard key

onload : The browser has finised the loading page

6 Javascript Sorting Arrays

6.1 Sorting An Array

The sort() method sorts an array alphabetically

6.2 Reversing An Array

The reverse() method reverses the elements in an array

6.3 toSorted() Method

ES2023 added the toSorted() method as a safe way to sort an array without altering the original array.
The difference between toSorted() and sort() is that the first method creates a new array, keeping the original array unchanged, while the last method alters the original array.
Similarly **toReversed()** used to reverse an array without altering the original array.

6.4 Numeric Sort

By default, the sort() function sorts values as strings.

7 JS HTML DOM

7.1 Finding HTML Elements

- document.getElementById(id) - Find an element by id name
- document.getElementsByTagName(name) - Find elements by tag name
- document.getElementsByClassName(name) - Find elements by class name

7.2 Changing HTML Elements

- element.innerHTML = new html content - Change the inner HTML of an element

- element.attribute = new value - Change the attribute value of an HTML element
- element.style.property = new style - Change the style of an HTML element

7.3 Adding And Deleting Elements

- document.createElement(element) - Create an HTML element
- document.removeChild(element) - Remove an HTML element
- document.appendChild(element) - Add an HTML element
- document.replaceChild(new, old) - Replace an HTML element
- document.write(text) - Write into the HTML output stream

7.4 Adding Event Handlers

- document.getElementById(id).onclick = function()code - Adding event handler code to an onclick event

7.5 Finding HTML Elements

7.5.1 By Id

Example: const element = document.getElementById("intro");
If the element is found, the method will return the element as an object (in element).
If the element is not found, element will contain null.

7.5.2 By Tag Name

Example: const element = document.getElementsByTagName("p"); This finds all <p> elements.

7.5.3 By Class Name

const x = document.getElementsByClassName("intro");
This example returns a list of all elements with class="intro".

7.6 Some related Information

7.6.1 Onload Event

The **onload** event can be used to check the visitor's browser type and browser version, and load the proper version of the web page based on the information.

7.6.2 Onunload Event

The onload and onunload events can be used to deal with cookies.
The onload and onunload events are triggered when the user enters or leaves the page

7.6.3 Oninput Event

The **oninput** event is often to some action while the user input data.
Example: <input type="text" id="fname" oninput="upperCase()">

7.6.4 Onchange Event

The **onchange** event is often used in combination with validation of input fields.

7.6.5 onmouseover and onmouseout Events

The **onmouseover** and **onmouseout** events can be used to trigger a function when the user mouses over, or out of, an HTML element