DTA Training

Arrays in JS & Date & Time Functions

By Deepika Singh

Agenda

Arrays

- Arrays Definition
- Array Creation
- Array Accessing
- Array Modifying
- Array Iteration
- For Each Loop
- Array Methods
- Multidimensional Array

Date Time Methods

- Date Object
- Date Get Methods
- Date Set Methods
- Date Formatting, Comparisions

Array’s Definition

An array is a data structure in JavaScript used to store and manage a collection of values.
Arrays can hold various data types, such as numbers, strings, objects, and even other arrays.

Creating Arrays

Using array literals:

```
const myArray = [1, 2, 3, 4, 5];
```

Using the new Array() constructor:

```
const myArray = new Array(1, 2, 3, 4, 5);
```

Back to Agenda Page

Accessing Array Elements

- Array elements are accessed by their index.
- Arrays are zero-indexed, meaning the first element is at index 0, the second at index 1, and so on.
- Example: const myArray = [10, 20, 30];
  console.log(myArray[0]);
  // Output: 10

Modifying Array Elements

You can modify array elements by assigning new values to them.

Example: const myArray = [10, 20, 30];

myArray[1] = 25;

Array Length

- You can find the number of elements in an array using the length property.
- Example: const myArray = [10, 20, 30];
  console.log(myArray.length);
  // Output: 3

Adding and Removing Elements

- You can add and remove elements using various methods, such as push(), pop(), shift(), and unshift().
- Example: const myArray = [1, 2, 3]; myArray.push(4);

Iterating Through Arrays

- You can loop through array elements using for loops, forEach(), or other looping constructs.
- Example: Using a for loop to iterate through an array.

For Each loop

A forEach loop allows you to iterate over the elements of an array or array-like object and perform a specified action (a function) on each element.

Syntax :

```
array.forEach(function(currentValue, index, array)
 {
  // Your code to process each element goes here
});
```

- array: The array you want to iterate over.
- currentValue: The current element being processed.
- index (optional): The index of the current element.
- array (optional): The array that forEach was called upon.

Array Methods

- JavaScript provides a wide range of array methods to manipulate arrays efficiently.
- Common methods include map(), filter(), reduce(), and more.

Back to Agenda Page

Multidimensional Arrays

- JavaScript allows you to create arrays of arrays, forming multidimensional arrays.
- Example: const matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]];

- Accessing Elements : :matrix[1][0]
- console.log(element); // Output: 5

- Modifying Elements : :matrix[1][0] = 12

Iterating Over a Multidimensional Array

```
for (let i = 0; i < twoDimArray.length; i++) {
  for (let j = 0; j < twoDimArray[i].length; j++) {
    console.log(`Element at row ${i}, column ${j}:
${twoDimArray[i][j]}`);
  }
}
```

Date and Time Methods in JavaScript

- Managing date and time is a fundamental aspect of many web applications, and JavaScript provides robust tools to handle these effectively.
- In this presentation, we'll explore JavaScript's Date object and various methods for working with dates and times.

Back to Agenda Page

The Date Object

- In JavaScript, the Date object represents a date and time.
- You can create a new Date object using the new Date() constructor.
- By default, it initializes with the current date and time.

Creating Date objects

- Current date and time: const currentDate = new Date();
- Specific date and time: const customDate = new Date('2023-11-01T12:00:00');

Date Methods - Getting Information

Method	Description
getFullYear()	Get year as a four digit number (yyyy)
getMonth()	Get month as a number (0-11)
getDate()	Get day as a number (1-31)
getDay()	Get weekday as a number (0-6)
getHours()	Get hour (0-23)
getMinutes()	Get minute (0-59)
getSeconds()	Get second (0-59)
getMilliseconds()	Get millisecond (0-999)
getTime()	Get time (milliseconds since January 1, 1970)

Date Methods - Setting Information

Method	Description
setDate()	Set the day as a number (1-31)
setFullYear()	Set the year (optionally month and day)
setHours()	Set the hour (0-23)
setMilliseconds()	Set the milliseconds (0-999)
setMinutes()	Set the minutes (0-59)
setMonth()	Set the month (0-11)
setSeconds()	Set the seconds (0-59)
setTime()	Set the time (milliseconds since January 1, 1970)

Formatting Dates

JavaScript provides methods to format dates into strings, such as:

- .toDateString()
- .toLocaleDateString()
- .toLocaleTimeString()
- .toLocaleString()

Comparing Dates

You can compare dates using various comparison operators, such as >, <, ===, and the methods .getTime() and .getTimezoneOffset().

Performing Date Arithmetic

JavaScript allows you to perform arithmetic operations with dates.
You can add or subtract time intervals to/from dates.
Example: Calculating the date one week from today.

Thank You