Module 5: Introduction to JavaScript

Introduction to JavaScript

- JavaScript, a core web technology alongside HTML and CSS, enables dynamic content creation on websites.
- It is an interpreted, single-threaded, multi-paradigm language with dynamic typing, supporting imperative, object-oriented, and functional programming.
- Classic JavaScript runs in web browsers, primarily for frontend development, while modern JavaScript (e.g., Node.js) supports full-stack development and multithreading with web workers.
- JavaScript utilises variables, literals, operators, expressions, and control flow statements, with code written in .js files terminated by semicolons.
- Debugging is done via IDE tools for non-browser environments or developer tools in browsers, supporting responsive and cross-platform application development.

Variables, Data Types and Operators in JavaScript

- Variables in JavaScript are symbolic names used to reference data, including user input, intermediate values, and final results.
- An identifier (variable name) must follow four rules: it can only contain letters, digits, underscores, and dollar signs; it must begin with a letter, underscore, or dollar sign; it is case-sensitive; and reserved words cannot be used.
- Variables in JavaScript use dynamic typing, meaning the data type is not declared, and it changes based on the value assigned. The typeof operator can be used to check the data type.
- JavaScript supports seven arithmetic operators, increment and decrement operators, and assignment shorthand operators like x += 8. Relational and logical operators are used for comparisons and building Boolean expressions.
- Operator precedence determines the evaluation order of operators, with higher precedence operators evaluated first. Parentheses can be used to override default precedence.

Conditional Control Flow with JavaScript

- JavaScript supports two conditional control flow statements: the if and switch statements.
- The if statement evaluates a Boolean expression to decide which statements to execute.
- The Boolean expression can be a literal, a relational comparison, or a combination of relational and logical operators.
- The if statement can include an else clause for alternative actions and else if for multiple conditions.
- The switch statement evaluates a discrete value and executes the corresponding case, with a break to prevent fall-through.

Iterative Control Flow with JavaScript

- Iterative control flow statements in JavaScript allow repetitive execution of code blocks, using loops like while, do while, and for.
- A while loop executes as long as the condition is true, but can result in an infinite loop if the condition never becomes false.
- A do while loop ensures the body runs at least once before evaluating the condition.
- A for loop combines initialisation, condition check, and increment, repeating code based on the counting variable.
- A break and continue statements allow premature loop termination or skipping to the next iteration, respectively.

JavaScript Functions and Scope

- JavaScript functions modularise code, improving maintainability and reusability across different files.
- A function is defined with the function keyword, followed by a name, parameters (if any), and a block of code in curly brackets.
- JavaScript uses dynamic typing, allowing functions to accept parameters of different data types and return corresponding values.
- Functions are invoked by their name with parentheses, and execution stops at the return statement, sending a value back to the caller.
- Variables can have local or global scope, with local variables accessible only within their respective functions.

Basic Data Structures in JavaScript

- Data structures in JavaScript, like arrays and objects, store multiple values under one variable, unlike a single-value variable.
- Arrays can hold multiple values, referenced by a zero-based index, and are dynamic and resizable.
- JavaScript arrays offer methods like push and pop for easy manipulation, with the length property returning the array size.
- Objects store name-value pairs and can be accessed using square bracket or dot notation.
- Arrays and objects can be iterated with loops, and multidimensional structures like arrays of objects are also common.

Overview of DOM Programming Interface

- The Document Object Model (DOM) is created by the browser when a web page is loaded, representing HTML elements as objects in a tree structure.
- The DOM can be modified using the browser developer tools to change the structure and content of a webpage.
- The DOM provides a programming interface, where HTML elements are objects with properties, methods, and events.

- Properties of DOM elements can be read and modified, methods can perform actions like adding or deleting elements, and events allow user interactions.
- Using JavaScript and the DOM, dynamic HTML content can be created and modified within the browser.

Basic DOM Manipulation with JavaScript

- JavaScript interacts with the Document Object Model (DOM) to create dynamic HTML content in web browsers.
- To manipulate DOM objects, you first obtain a reference using methods like getElementById, getElementsByTagName, or getElementByClassName.
- JavaScript allows you to modify properties such as innerHTML and CSS styles, and add or remove DOM objects dynamically.
- Event handling in the DOM is managed through addEventListener and removeEventListener, supporting event bubbling and capturing.
- The DOM tree is hierarchical, allowing navigation between nodes using properties like parentNode and childNodes for efficient manipulation.

THE END

My notes...