Lesson 4: JavaScript Basics

JavaScript is a high-level, interpreted programming language primarily used for web development. It enables developers to add interactive and dynamic elements to websites, such as validating forms, creating animations, manipulating data, and much more. JavaScript is supported by all modern web browsers, making it an essential tool for front-end development.

JavaScript Syntax:

To write and run JavaScript code, you need a text editor and a web browser. Popular text editors like Visual Studio Code, Sublime Text, or Atom are great choices. You can create a new file with a ".js" extension, such as "script.js," to save your JavaScript code.

To execute your JavaScript code, you can either embed it directly in an HTML file using the <script> tag or link an external JavaScript file using the `<script src="script.js"></script>` tag within the HTML document. Open the HTML file in your web browser, and the JavaScript code will be executed.

JavaScript has a C-like syntax with a combination of curly braces, parentheses, and semicolons.

Here's an example of a simple JavaScript program that displays a message in the console:

```
// Declare a variable
var message = "Hello, World!";

// Display the message in the console
console.log(message);
```

In JavaScript, **variables** are declared using the var, let, or const keywords. Variables declared with var and let can be reassigned, while variables declared with const are read-only and cannot be reassigned. It's good practice to use const whenever possible to ensure immutability.

Here are some important aspects of JavaScript syntax:

Statements: JavaScript code is composed of statements, which are instructions or actions. Each statement is terminated by a semicolon (;). For example:

```
var x = 5;
```

Variables: JavaScript uses variables to store data values. Variables are declared using the var, let, or const keywords, followed by a variable name. For example:

```
var message = "Hello, world!";
```

Variables can hold various types of data, and their values can be changed during the execution of a program.

Case Sensitivity: JavaScript is case-sensitive, meaning that myVariable and myvariable are treated as different variables.

Data Types in JavaScript:

JavaScript supports several data types, including numbers, strings, booleans, objects, arrays, and more.
Here are some examples:

```
Numbers: let age = 25;
Strings: let name = "John";
Booleans: let isStudent = true;
```

1. **Numbers:** The number data type represents numeric values, including integers and floating-point numbers. For example:

```
var age = 25;
var pi = 3.14;
```

2. **Strings:** Strings are sequences of characters enclosed in single (') or double ("") quotes. They are used to represent text or a series of characters. For example:

```
var name = "John";
var message = 'Hello, world!';
```

3. **Booleans:** The boolean data type represents logical values. It can only have two possible values: true or false. Booleans are often used in conditional statements and comparisons. For example:

```
var isLogged = true;
var isValid = false;
```

4. **Arrays:** Arrays are ordered collections of values enclosed in square brackets ([]). They can hold multiple values of any data type. Array elements are accessed using zero-based indexing. For example:

```
var numbers = [1, 2, 3, 4, 5];
var fruits = ["apple", "banana", "orange"];
```

5. **Objects:** Objects are collections of key-value pairs enclosed in curly braces ({}). They are used to represent complex data structures and can contain properties and methods. For example:

```
var person = {
  name: "John",
  age: 30,
  city: "New York"
};
```

6. **Null and Undefined:** null represents the intentional absence of any object value. It is used to indicate that a variable has no value or an empty value. undefined is a special value assigned to variables that have been declared but not yet assigned a value.

Operators in Javascript:

JavaScript also provides various operators, such as arithmetic operators (**+, -, *, /**), assignment operators (**=, +=, -=**), comparison operators (**==, ===, !=, !==, <, >, <=, >=**), and logical operators (**&&, ||, !**).

1. **Arithmetic Operators:** Arithmetic operators perform mathematical calculations on numeric values. Examples include addition **(+)**, subtraction **(-)**, multiplication **(*)**, division **(/)**, and modulus **(%)**.

2. **Assignment Operators:** Assignment operators assign values to variables. Examples include the assignment operator **(=)**, as well as compound assignment operators like **+=, -=, *=, /=**, which perform the operation and assign the result to the variable in one step.

3. **Comparison Operators:** Comparison operators compare two values and return a Boolean result (true or false). Examples include equality (**==** and **===**), inequality (**!=** and **!==**), greater than **(>)**, less than **(<)**, greater than or equal to **(>=)**, and less than or equal to **(<=).**

4. **Logical Operators:** Logical operators are used to combine or negate Boolean values. Examples include the logical AND (**&&**), logical OR **(||)**, and logical NOT **(!)** operators.

5. **String Operators:** The concatenation operator **(+)** is used to concatenate strings together. For example:

```
let greeting = "Hello, ";
let name = "John";
let message = greeting + name;
```

6. **Type Operators:** JavaScript provides operators to determine the type of a value. The typeof operator returns a string representing the type of the operand. For example:

```
typeof 42; // returns "number"
typeof "Hello"; // returns "string"
```

These are just a few examples of the many operators available in JavaScript. Operators allow you to perform calculations, comparisons, logical operations, and manipulate data of different types.

Understanding data types and operators is crucial as they form the building blocks for writing JavaScript code and manipulating data effectively.