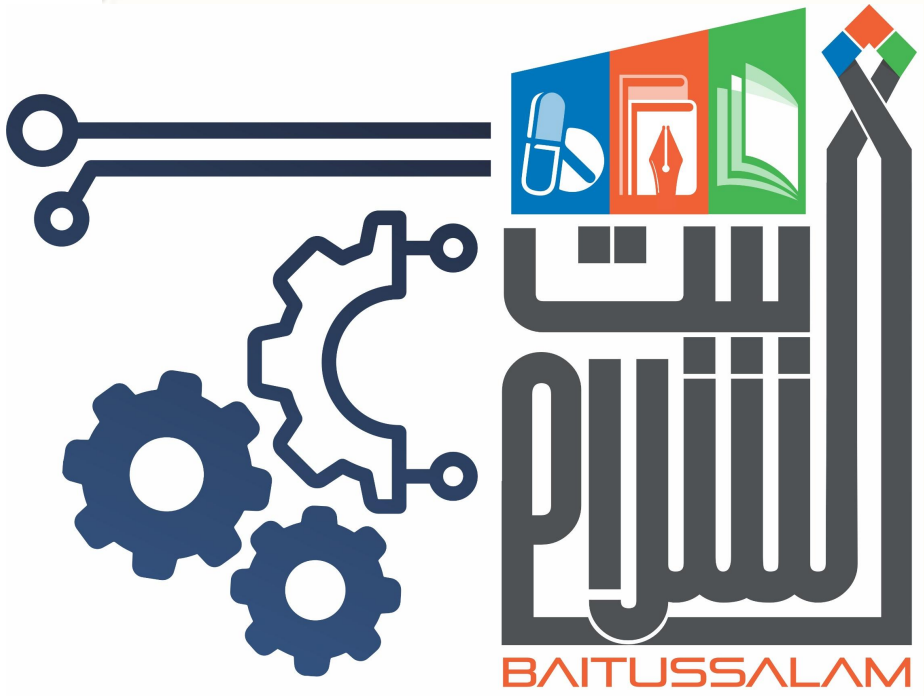

BAITUSSALAM
—TECH PARK—

BAITUSSALAM
—TECH PARK—

PSDC-201

Introduction to programming with Javascript and Typescript

Introduction to Javascript

BAITUSSALAM
—TECH PARK—

Prerequisite to learn Javascript

- HTML
- CSS
- Basic Programming Concepts

Interpreting JavaScript

JavaScript: A programming language used to add interactivity and dynamic behavior to web pages.

Browser interprets JavaScript code to handle user interactions, animations, and other dynamic features.

Examples: Form validation, image sliders, interactive games.

```
// Sample array
const colors = ["red", "green", "blue", "yellow"];

// String to check for
const searchString = "blue";

// Using includes method
if (colors.includes(searchString)) {
  console.log(`${searchString} is in the array.`);
} else {
  console.log(`${searchString} is not in the array.`);
}
```

What is JavaScript ?

- JavaScript is a high-level, interpreted programming language.
- It's commonly used for creating interactive effects within web browsers.

Role of JavaScript in Web Development

Enhanced User Experience:

JavaScript allows for dynamic content updates without reloading the entire webpage.

Interactive elements like forms, animations, and pop-ups enhance user engagement.

Client-Side Scripting:

JavaScript runs on the client side, meaning it's executed by the user's browser.

This reduces server load and improves website performance by offloading tasks to the client's machine.

Role of JavaScript in Web Development

DOM Manipulation:

The Document Object Model (DOM) represents the structure of a webpage.

JavaScript enables developers to manipulate DOM elements, altering their content, style, and behavior.

Asynchronous Programming:

JavaScript supports asynchronous operations, allowing tasks to execute independently without blocking other processes.

This is crucial for handling events, fetching data from servers, and ensuring smooth user interactions.

Role of JavaScript in Web Development

Cross-Browser Compatibility:

JavaScript code is supported by all major web browsers, including Chrome, Firefox, Safari, and Edge.

This ensures consistency in the behavior and appearance of web applications across different platforms.

Rich Ecosystem of Libraries and Frameworks:

JavaScript has a vast ecosystem of libraries (e.g., jQuery, React) and frameworks (e.g., Angular, Vue.js) that streamline development.

These tools provide pre-built components, modules, and utilities to expedite the creation of complex web applications.

Javascript Syntax and Data Types

There are two different types of data types in javascript:

- **Primitive data type**

  Primitive data types in JavaScript are like different kinds of puzzle pieces that computers use to understand and handle information, such as numbers, words, yes or no questions, empty spaces, and things that haven't been figured out yet.

- **Non-primitive data type**

  Non-primitive data types in JavaScript are like special helpers that make computers do tricky things, like sorting and following instructions step by step.

Variable And Its Scope

In JavaScript, **var**, **let**, and **const** are used to declare variables, but they have different behaviors and scoping rules.

var v = 10;

let a = 15;

const = 50;

Variable And Its Scope

var:

- var was the original way to declare variables in JavaScript.
- Variables declared with var have function scope or global scope, but not block scope.
- This means that variables declared with var are accessible anywhere within the function they are declared in, or globally if declared outside any function.
- Variables declared with var can be re-declared and updated within their scope.

```
function example() {
    var x = 10;
    if (true) {
        var x = 20; // This will overwrite
        console.log(x); // Output: 20
    }
    console.log(x); // Output: 20
}
```

Variable And Its Scope

let:

- *let* was introduced in ES6 (ECMAScript 2015) to provide block-scoped variables.
- Variables declared with *let* have block scope, meaning they are only accessible within the block they are declared in (e.g., within curly braces {}).
- *let* variables can be reassigned, but not re-declared within the same scope.
- *let* declarations are not hoisted to the top of their block, unlike var.

- Using let can help prevent certain types of bugs and make your code more predictable.

```
function example() {
    let x = 10;
    if (true) {
        let x = 20; // This creates a new variable
        console.log(x); // Output: 20
    }
    console.log(x); // Output: 10
}
```

Variable And Its Scope

const:

- const is also introduced in ES6 and is used to declare constants.
- Variables declared with const must be initialized with a value and cannot be reassigned or re-declared.
- const variables also have block scope.
- However, note that when using const with complex data types like arrays or objects, the variable itself is immutable (cannot be reassigned), but the content of the data structure can still be modified.

```
function example() {
    const x = 10;
    // x = 20; // This will cause an error
    console.log(x); // Output: 10
}
```

Javascript Syntax and Data Types

- **Primitive Data Type**
  - *String*
  - *Boolean*
  - *Number*
  - *Undefined*
  - *Null*

- **Non-Primitive Data Type**
  - *Object*
  - *Array*
  - *Function*

Primitive Data Types

1. **Numbers**:
Numbers are like the digits you use for counting or doing math. They can be big or small, positive or negative, and even have decimal points. For example, if you want to know how many cookies you have, you'd use a number data type.

2. **Strings**:
Strings are like words or sentences. They're made up of letters, numbers, or symbols. So, if you want to write your name or a message to your friend, you'd use a string data type.

3. **Booleans**:
Booleans are like yes or no questions. They can only have two values: true or false. For instance, if you want to know if it's raining outside, you'd use a boolean data type. If it's raining, it's true; if it's not raining, it's false.

4. **Null**:
Null means nothing or empty. It's like having an empty box. There's nothing inside it. Sometimes, programmers use null when they want to say that something doesn't exist or has no value.

5. **Undefined**:
Undefined means something hasn't been defined or set yet. It's like having an empty space where something should be, but it hasn't been put there yet. It's used when we forget to give something a value or when it's not clear what the value should be.

Non Primitive Data Types

1. **Objects**:
Objects are like treasure chests that can hold many different types of things inside them. They can store numbers, words, and even other objects! For example, an object could represent your toy collection, with each toy having its own name and color.

2. **Arrays**:
Arrays are like special containers that can hold multiple items in a list. They're like shelves with lots of compartments where you can organize things. For instance, an array could store your favorite snacks in order, from first to last.

3. **Functions**:
Functions are like magic spells that perform specific tasks when called upon. They're like recipes that tell the computer what steps to follow. For example, a function could be a recipe for making your favorite sandwich, with instructions on how to put it together.

Javascript Variables

Number

let length = 10;

let width = 20;

String

let firstName = “ahmed”;

Let lastName = “ali”;

Boolean

let x = true;

Let y = false;

Object

const person = { “firsName” : “Ali”, “lastName” : “Hasan”} ;

Array Object

const colors = [“Red”, “Blue”, “Green”, “Yellow”];

Date Object

const date = new Date(“2024-03-21”);

Javascript Math Operators

Addition (+): Adds two numbers together.

Example: 3 + 5 equals 8.

Subtraction (-): Subtracts one number from another.

Example: 7 - 4 equals 3.

Multiplication (*): Multiplies two numbers.

Example: 2 * 6 equals 12.

Division (/): Divides one number by another.

Example: 10 / 2 equals 5.

Modulus (%): Returns the remainder of a division operation.

Example: 10 % 3 equals 1, because 10 divided by 3 equals 3 with a remainder of 1.

Exponentiation ()**: Raises one number to the power of another.

Example: 2 ** 3 equals 8, because 2 raised to the power of 3 equals 8.

Javascript Code Execution

Everything inside javascript happens inside **Execution Context.**

Execution Context consist of two blocks

- **Memory** (variable environment)
- **Code** (Thread of Execution)

Memory Block

It is also called **variable environment.** It initializes all the declared variables or functions etc with undefined value in the memory. It keeps the data in the form of key value format.

Key : Value

a : 10

fn : {....}

Code Block

It is also called **Thread of Environment**. It executes all the line one by one and therefore we called javascript synchronous single-threaded language.

The End

BAITUSSALAM
—TECH PARK—