Introduction to JavaScript Basics

Title: JavaScript Basics and Node.js Setup
Instructor: Pramod Kumar Jena

---

1. Setting Up JavaScript Development Environment

Objective:
Introduce students to the development setup, ensuring they can run JavaScript code in VS Code.

Steps:

1. **Install Node.js:**
   Download and install Node.js from nodejs.org.
   - Ensure Node.js and npm are installed correctly by typing `node -v` and `npm -v` in the terminal.
   - Node.js allows students to run JavaScript code on their local machines.
2. **Set Up VS Code:**
   - Install Visual Studio Code.
   - Show them how to set up a new project folder.
   - Create a file called `index.js` (JavaScript file).
   - Show how to use the terminal in VS Code to run a JavaScript file using `node index.js`.

Exercise:

Run the following code to verify the setup:

```
console.log("Hello, JavaScript World!");
```

---

2. JavaScript Variables and Constants

Objective:
Introduce the concepts of variables and constants using `var`, `let`, and `const`.

2.1. What is a Variable?

A variable is a container where you can store data for use later. In JavaScript, variables can store different data types like numbers, strings, or booleans.

2.2. Declaring Variables: `var`, `let`, and `const`

- **`var`:** Function-scoped variable, can be reassigned.
- **`let`:** Block-scoped, can be reassigned.
- **`const`:** Block-scoped, cannot be reassigned (used for constants).

Example:

```
var age = 25;
let name = "John";
const birthYear = 1995;
```

2.3. Real-Life Example

Imagine storing the details of a person in a system:

```
let personName = "Alice";  // A variable can change, e.g., someone
updating their name.
const birthDate = "2000-01-01";  // Birthdate remains constant.
```

Exercise:

- Declare variables for `userName`, `userAge`, and `userCity`.
- Create a constant for the `birthYear`.

---

3. Keywords: `let`, `var`, `const`

3.1. Scope Differences:

- **Block scope (`let` and `const`):** Accessible only within a block `{}`.
- **Function scope (`var`):** Accessible within the function or globally if declared outside a function.

Example:

```
if (true) {
  let blockScoped = "Inside block";
```

```
  var functionScoped = "Inside block";
}
console.log(functionScoped);  // Works
console.log(blockScoped);     // Error: blockScoped is not defined
```

3.2. Reassignment Behavior:

- Variables declared with `let` and `var` can be reassigned.
- Variables declared with `const` cannot be reassigned.

---

4. Naming Conventions

Objective:
Introduce different naming conventions used in JavaScript.

4.1. Camel Case:

First word is lowercase, and subsequent words are capitalized.
`myVariableName`

4.2. Pascal Case:

All words are capitalized.
`MyVariableName`

4.3. Snake Case:

Words are separated by underscores.
`my_variable_name`

---

5. Basic JavaScript Syntax and Printing Output

Objective:
Understand the basic syntax and how to display output.

5.1. Syntax Rules:

- JavaScript is case-sensitive (`age` is not the same as `Age`).
- Variables must be declared before use.

5.2. Printing Output:

- `console.log()` is used to display output in the console.

Example:

```
let message = "Hello, World!";
console.log(message);  // Output: Hello, World!
```

Exercise:

- Ask students to declare a variable `greeting` and print it using `console.log()`.

---

6. JavaScript Operators

Objective:
Explore arithmetic, assignment, comparison, and logical operators.

6.1. Arithmetic Operators:

- Addition (`+`)
- Subtraction (`-`)
- Multiplication (`*`)
- Division (`/`)

Example:

```
let sum = 5 + 10;  // 15
```

6.2. Assignment Operators:

- Assign (`=`), Add and assign (`+=`), Subtract and assign (`-=`)

6.3. Comparison Operators:

- Equal to (`==`), Strict equal to (`===`)
- Greater than (`>`) or less than (`<`)

6.4. Logical Operators:

- AND (`&&`), OR (`||`), NOT (`!`)

Example:

```
let isAdult = age >= 18;
let hasID = true;
console.log(isAdult && hasID);  // true
```

Exercise:

Use comparison and logical operators to evaluate a person's age and print if they are eligible to vote.

---

7. Data Types in JavaScript

7.1. Primitive Data Types:

- **String:** `"Pramod"`
- **Number:** `25`, `3.14`
- **Boolean:** `true`, `false`
- **Undefined:** Variable declared but not assigned.
- **Null:** Represents intentional absence of any object value.

7.2. Non-Primitive Data Types:

- **Array:** A list of values.
- **Object:** Key-value pairs.