INTERNSHIP REPORT

WEEK 4 DAY 2

Submitted to:	**Ali Hyder**	Submission Date:	**15th July, 2025**
Internship Domain:	**Front Development**	Internship Name:	**ProSensia**
Student Name:	**Yasal Qamar**	Roll No.	**S25031**

JavaScript Variables and Data Types

Objective

To learn and practice JavaScript fundamentals by exploring how to declare variables using var, let, and const, and to understand the difference between **Primitive** and **Non-Primitive** data types in JavaScript.

Introduction:

1. JavaScript Variables

🔷 **var**

- Function-scoped.
- Allows redeclaration.
- Can be reassigned.
- Hoisted (declared before execution).

🔷 **let**

- Block-scoped.
- Cannot be redeclared in the same scope.
- Can be reassigned.
- Preferred in modern JavaScript.

🔷 **const**

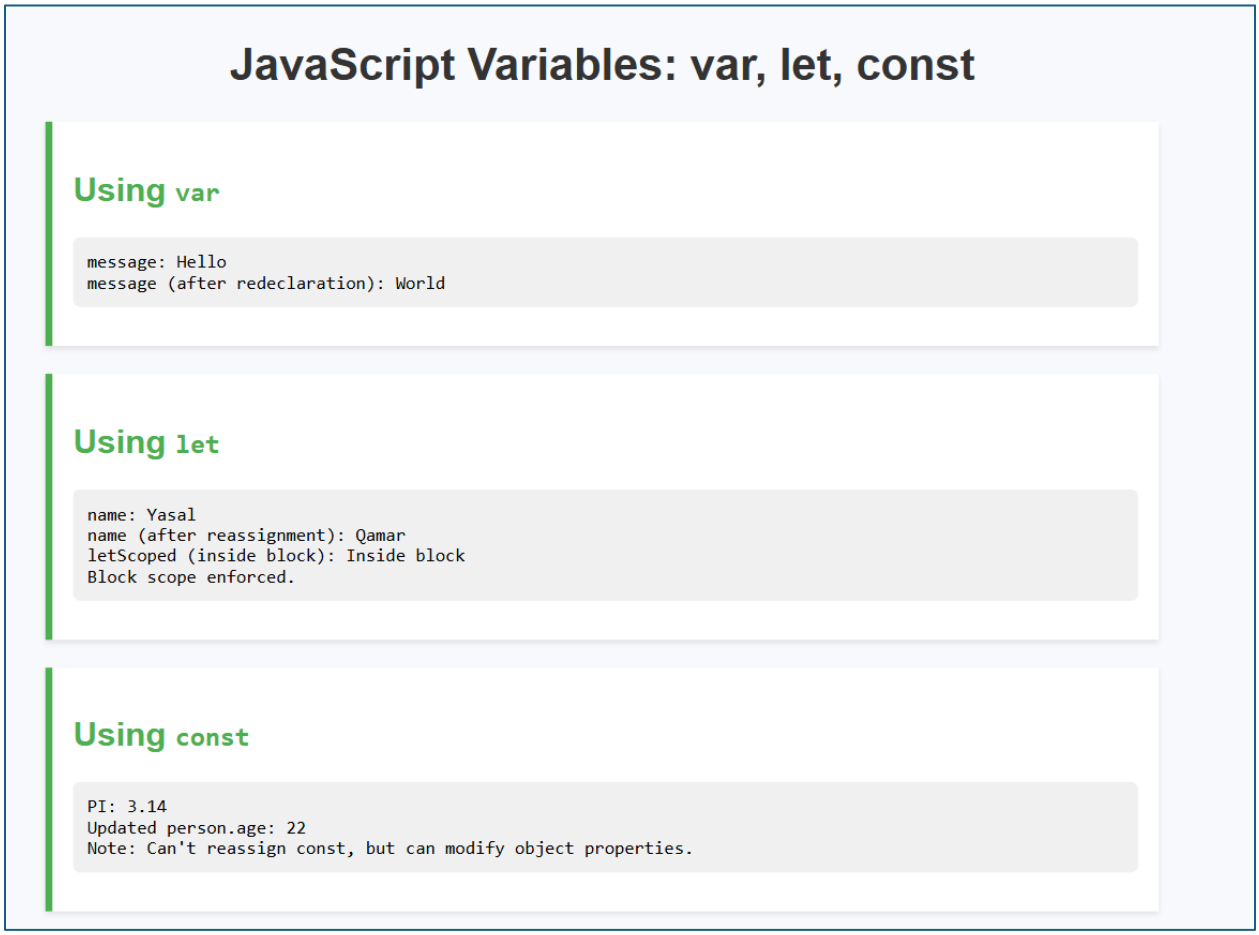
- Block-scoped.
- Cannot be redeclared or reassigned.

CODING

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>JavaScript Variables Example</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f7f9fc;
      padding: 20px;
      max-width: 800px;
      margin: auto;
    }

    h1 {
      text-align: center;
      color: #333;
    }

    .section {
      background-color: #ffffff;
      padding: 15px;
      margin-bottom: 20px;
      border-left: 5px solid #4CAF50;
      box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    }

    .section h2 {
      color: #4CAF50;
    }

    pre {
      background-color: #f0f0f0;
      padding: 10px;
      overflow-x: auto;
      border-radius: 5px;
    }
  </style>
</head>
<body>

  <h1>JavaScript Variables: var, let, const</h1>
```

```
<div class="section">
  <h2>Using <code>var</code></h2>
  <pre id="varOutput"></pre>
</div>

<div class="section">
  <h2>Using <code>let</code></h2>
  <pre id="letOutput"></pre>
</div>

<div class="section">
  <h2>Using <code>const</code></h2>
  <pre id="constOutput"></pre>
</div>

<!-- ✅ JavaScript Starts Here -->
<script>
  // VAR example
  var message = "Hello";
  let varOutput = "message: " + message + "\n";
  var message = "World";
  varOutput += "message (after redeclaration): " + message;

  document.getElementById("varOutput").textContent = varOutput;

  // LET example
  let name = "Yasal";
  let letOutput = "name: " + name + "\n";
  name = "Qamar";
  letOutput += "name (after reassignment): " + name + "\n";

  {
    let letScoped = "Inside block";
    letOutput += "letScoped (inside block): " + letScoped + "\n";
  }
  letOutput += "Block scope enforced.";

  document.getElementById("letOutput").textContent = letOutput;

  // CONST example
  const PI = 3.14;
  let constOutput = "PI: " + PI + "\n";

  const person = { name: "Yasal", age: 21 };
```

```
    person.age = 22;
    constOutput += "Updated person.age: " + person.age + "\n";

    constOutput += "Note: Can't reassign const, but can modify object
properties.";

    document.getElementById("constOutput").textContent = constOutput;
  </script>

</body>
</html>
```



2. JavaScript Data Types

🔶 **Primitive Types:**

- **String** – Text data ("Yasal")
- **Number** – Numeric values (21)

- **Boolean** – True/false values
- **Undefined** – Declared but not assigned
- **Null** – Explicitly no value

🔶 **Non-Primitive (Reference) Types:**

- **Object** – Key-value pairs (e.g. {name: "Yasal"})
- **Array** – List of items (["Red", "Green", "Blue"])
- **Function** – Reusable block of code

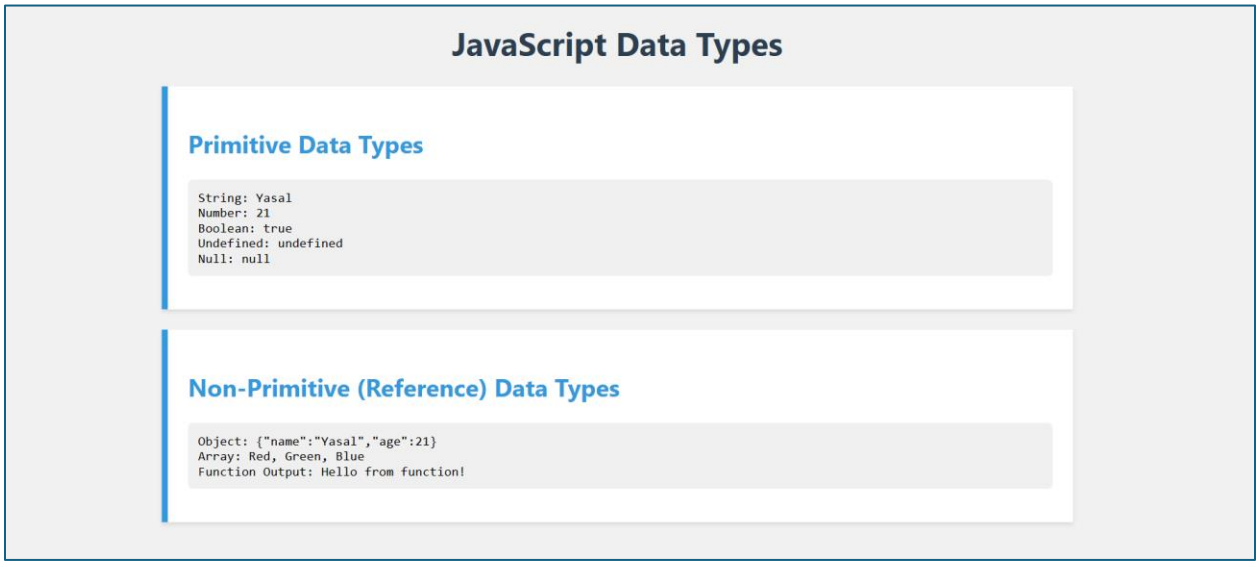
CODING

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>JavaScript Data Types</title>
  <style>
    body {
      font-family: 'Segoe UI', sans-serif;
      background: #f1f1f1;
      padding: 30px;
      max-width: 900px;
      margin: auto;
    }

    h1 {
      text-align: center;
      color: #2c3e50;
    }

    .section {
      background-color: #fff;
      padding: 20px;
      margin: 20px 0;
      border-left: 6px solid #3498db;
      box-shadow: 0 2px 4px rgba(0,0,0,0.1);
    }

    .section h2 {
```

```
      color: #3498db;
    }

    pre {
      background: #f0f0f0;
      padding: 10px;
      border-radius: 5px;
      overflow-x: auto;
    }
  </style>
</head>
<body>

  <h1>JavaScript Data Types</h1>

  <div class="section">
    <h2>Primitive Data Types</h2>
    <pre id="primitiveOutput"></pre>
  </div>

  <div class="section">
    <h2>Non-Primitive (Reference) Data Types</h2>
    <pre id="nonPrimitiveOutput"></pre>
  </div>

  <script>
    // ✅ Primitive Data Types
    let name = "Yasal";              // String
    let age = 21;                    // Number
    let isStudent = true;           // Boolean
    let notDefined;                 // Undefined
    let nothing = null;            // Null

    let primitiveOutput = "";
    primitiveOutput += "String: " + name + "\n";
    primitiveOutput += "Number: " + age + "\n";
    primitiveOutput += "Boolean: " + isStudent + "\n";
    primitiveOutput += "Undefined: " + notDefined + "\n";
    primitiveOutput += "Null: " + nothing + "\n";

    document.getElementById("primitiveOutput").textContent = primitiveOutput;

    // ✅ Non-Primitive Data Types
    let person = { name: "Yasal", age: 21 };   // Object
    let colors = ["Red", "Green", "Blue"];     // Array
```

```
    function greet() {                          // Function
      return "Hello from function!";
    }

    let nonPrimitiveOutput = "";
    nonPrimitiveOutput += "Object: " + JSON.stringify(person) + "\n";
    nonPrimitiveOutput += "Array: " + colors.join(", ") + "\n";
    nonPrimitiveOutput += "Function Output: " + greet();

    document.getElementById("nonPrimitiveOutput").textContent =
nonPrimitiveOutput;
  </script>

</body>
</html>
```

Output:



Category	Type	Example Used
Primitive	String	"Yasal"
	Number	21
	Boolean	true
	Undefined	let x;
	Null	null
Non-Primitive	Object	{ name: "Yasal" }
	Array	["Red", "Green"]
	Function	function greet() {...}

Key Learnings:

- Understanding scope and usage of var, let, and const improves code structure.
- JavaScript's dynamic typing allows flexibility but requires careful handling.
- Knowing data types is crucial for writing logical conditions, validations, and functions.

Tools Used:

- Code Editor: Notepad / VS Code
- Browser Console for Debugging
- HTML, CSS for UI structure
- JavaScript for scripting and output handling

Conclusion:

This session was highly productive in building my confidence in JavaScript fundamentals. By combining HTML, CSS, and JavaScript, I developed interactive web pages that clearly demonstrated variable behavior and data type usage an essential step in frontend development.