A Quick Dive into JavaScript

Introduction to JavaScript:

JavaScript, or JS, is a dynamic programming language widely used for web development. It enables developers to create interactive and dynamic user interfaces.

REPL in JS:
Explore JavaScript code instantly using a Read-Eval-Print Loop (REPL). Open your browser console or use online editors like CodePen to experiment and see immediate results.

Variables in JS:
Variables store data in JavaScript. Declare them using `var`, `let`, or `const`. Example: `let myVar = 42;`

Data Types in JS:
JavaScript supports various data types like numbers, strings, booleans, objects, arrays, null, and undefined, each serving a unique purpose.

Numbers in JS:
Handle integers and floating-point numbers effortlessly. Perform operations like addition, subtraction, multiplication, and division directly in your code.

```
let num1 = 10;
let num2 = 5;
let result = num1 + num2;
```

Operations in JS:
JavaScript offers a range of operations, including arithmetic, comparison, logical, and bitwise operations, crucial for writing effective code.

NaN in JS:
`NaN` represents an undefined or unrepresentable value in numeric computations. Be cautious when encountering it in your calculations.

```
let result = "Hello" / 2;
```

```
console.log(isNaN(result)); // true
```

Operator Precedence:
Understand the order in which operations execute in an expression to avoid unexpected outcomes.

```
let result = 10 + 5 * 2; // 20
```

let, const, and var Keywords:
Use `let` and `const` for block-scoped variables, and `var` for function-scoped. Choose based on your variable requirements.

Assignment Operators and Unary Operators:
Assign values with operators like `=`, and perform unary operations like increment (`++`) or decrement (`--`).

```
let x = 5;
x++; // 6
```

Boolean in JS:
Boolean values (`true` or `false`) are fundamental for control flow, conditionals, and logical operations.

```
let isTrue = true;
let isFalse = false;
```

String in JS:
Manipulate sequences of characters with string operations like concatenation, slicing, and length determination.

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

String Indices:
Access individual characters using indices starting from 0.

```
let word = "JavaScript";
```

console.log(word[0]); // J

null and undefined in JS:
Differentiate between intentional absence (`null`) and declared but unassigned variables (`undefined`) for error-free coding.
let nullValue = null;
let undefinedValue;
console.log(nullValue); // null
console.log(undefinedValue); // undefined

Check_out_my_detailed_blog:-
https://medium.com/@srivastavayushmaan1347/unveiling-the-world-of-javascript-9abf49dbd797