Introduction to JavaScript

Welcome to your JavaScript journey! This guide covers the fundamental building blocks of JavaScript programming.

Table of Contents

Variables

Variables are containers that store data values. Think of them as labeled boxes where you can put different types of information.

Declaration Methods

```
// let - for variables that can change
let age = 25;
age = 26; // This is allowed

// const - for variables that don't change
const name = "Alice";
// name = "Bob"; // This would cause an error

// var - older way (avoid in modern JavaScript)
var city = "New York";
```

Naming Rules

- Must start with a letter, underscore (_), or dollar sign ($)
- Cannot start with a number
- Case-sensitive (age and Age are different)
- Use camelCase for multiple words: `firstName`, `totalPrice`

Data Types

JavaScript has several built-in data types. Here are the main ones:

Primitive Types

1. Number

```
let integer = 42;
let decimal = 3.14159;
let negative = -17;
```

2. String

```
let singleQuotes = 'Hello World';
let doubleQuotes = "JavaScript is fun";
let templateLiteral = `My age is ${age}`; // Can include variables
```

3. Boolean

```
let isStudent = true;
let isGraduated = false;
```

4. Undefined

```
let notAssigned;
console.log(notAssigned); // undefined
```

5. Null

```
let emptyValue = null; // Intentionally empty
```

Checking Data Types

```
console.log(typeof 42);        // "number"
console.log(typeof "hello");   // "string"
console.log(typeof true);      // "boolean"
```

Operators

Operators perform operations on variables and values.

Arithmetic Operators

```

```

```
let a = 10;
let b = 3;

console.log(a + b);  // 13 (addition)
console.log(a - b);  // 7  (subtraction)
console.log(a * b);  // 30 (multiplication)
console.log(a / b);  // 3.33... (division)
console.log(a % b);  // 1  (remainder/modulus)
console.log(a ** b); // 1000 (exponentiation)
```

Assignment Operators

```
let x = 5;
x += 3;  // Same as: x = x + 3 (now x is 8)
x -= 2;  // Same as: x = x - 2 (now x is 6)
x *= 2;  // Same as: x = x * 2 (now x is 12)
x /= 4;  // Same as: x = x / 4 (now x is 3)
```

Comparison Operators

```
let num1 = 5;
let num2 = "5";

console.log(num1 == num2);   // true (loose equality)
console.log(num1 === num2);  // false (strict equality)
console.log(num1 != num2);   // false (loose inequality)
console.log(num1 !== num2);  // true (strict inequality)
console.log(num1 > 3);       // true
console.log(num1 <= 5);      // true
```

Logical Operators

```
let hasLicense = true;
let hasInsurance = false;

console.log(hasLicense && hasInsurance); // false (AND)
console.log(hasLicense || hasInsurance); // true (OR)
console.log(!hasLicense);                // false (NOT)
```

Control Flow

Control flow statements determine the order in which code executes.

If Statements

```
let temperature = 75;

if (temperature > 80) {
    console.log("It's hot outside!");
} else if (temperature > 60) {
    console.log("Nice weather today!");
} else {
    console.log("It's cold outside!");
}
```

Switch Statements

```
let dayOfWeek = "Monday";

switch (dayOfWeek) {
    case "Monday":
        console.log("Start of work week");
        break;
    case "Friday":
        console.log("TGIF!");
        break;
    case "Saturday":
    case "Sunday":
        console.log("Weekend!");
        break;
    default:
        console.log("Regular day");
}
```

Loops

For Loop

```
// Print numbers 1 to 5
for (let i = 1; i <= 5; i++) {
    console.log(i);
}
```

While Loop

```
let count = 0;
while (count < 3) {
    console.log("Count is: " + count);
    count++;
}
```

For...of Loop (for arrays)

```
let colors = ["red", "green", "blue"];
for (let color of colors) {
    console.log(color);
}
```

Functions

Functions are reusable blocks of code that perform specific tasks.

Function Declaration

```
function greetUser(name) {
    return "Hello, " + name + "!";
}

let message = greetUser("Alice");
console.log(message); // "Hello, Alice!"
```

Function Expression

```
const calculateArea = function(width, height) {
    return width * height;
};

let area = calculateArea(5, 3);
console.log(area); // 15
```

Arrow Functions (Modern JavaScript)

```
// Short version for simple functions
const square = (num) => num * num;

// Longer version for complex functions
const calculateTip = (bill, tipPercent) => {
    const tip = bill * (tipPercent / 100);
    return bill + tip;
};

console.log(square(4));           // 16
console.log(calculateTip(50, 20)); // 60
```

Function Parameters and Arguments

```
// Default parameters
function introduce(name, age = 18) {
    return `Hi, I'm ${name} and I'm ${age} years old.`;
}

console.log(introduce("Bob"));        // Uses default age
console.log(introduce("Alice", 25));  // Uses provided age
```

Practice Exercises

Exercise 1: Variables and Data Types

Create variables for a student's information:

```
// Your solution here
let studentName = "Your Name";
let studentAge = 20;
let isEnrolled = true;
let gpa = 3.75;
```

Exercise 2: Calculator Function

Create a function that takes two numbers and an operator (+, -, *, /) and returns the result:

```
function calculate(num1, num2, operator) {
    // Your code here
}

// Test your function
console.log(calculate(10, 5, "+")); // Should return 15
```

Exercise 3: Grade Checker

Write a function that takes a score and returns a letter grade:

- 90-100: A
- 80-89: B
- 70-79: C
- 60-69: D
- Below 60: F

```
function getLetterGrade(score) {
    // Your code here
```

```
}
```

Exercise 4: Loop Practice

Write a function that prints all even numbers from 1 to a given number:

```
function printEvenNumbers(max) {
    // Your code here
}

printEvenNumbers(10); // Should print: 2, 4, 6, 8, 10
```

Key Takeaways

1. **Variables** store data using `let` for changeable values and `const` for constants
2. **Data types** include numbers, strings, booleans, undefined, and null
3. **Operators** perform arithmetic, comparison, and logical operations
4. **Control flow** uses if/else, switch, and loops to control program execution
5. **Functions** create reusable code blocks that can accept parameters and return values

Next Steps

Once you're comfortable with these concepts, you'll be ready to learn about JavaScript data structures like arrays, objects, and more advanced topics!

Happy coding! 🚀