JavaScript Fundamentals

A Complete Guide to Core Concepts

Instructor: Abhishek Garg

August 11, 2025

Contents

1 Console.log - Your Debugging Best Friend!

The `console.log()` function is like having a conversation with your computer. It displays information in the browser's console, making it essential for debugging and understanding your code.

1.1 Explanation

`console.log()` is a built-in JavaScript function that outputs data to the web console. It's primarily used for:

- Debugging code
- Displaying variable values
- Testing program flow
- Learning and experimentation

1.2 Examples

Example 1: Basic Usage

```javascript
console.log("Hello, World!");
console.log(42);
console.log(true);

// Multiple values
console.log("The answer is:", 42);
```

Output

```
Hello, World!
42
true
The answer is: 42
```

Example 2: Variables and Objects

```javascript
let studentName = "Alice";
let grade = 95;
let student = {name: "Bob", age: 20};

console.log("Student:", studentName);
console.log("Grade:", grade);
console.log("Student object:", student);
```

Output

```
Student: Alice
Grade: 95
Student object: {name: "Bob", age: 20}
```

1.3 Practice Questions

Practice Problems - Console.log

Problem 1: Pizza Order Tracker You're building a pizza ordering system. Use console.log to display order details.

```
// Your code here
// Display: customer name, pizza type, quantity, total price
```

Sample Input: Customer: "John", Pizza: "Margherita", Quantity: 2, Price per pizza: $12 **Expected Output:**

```
Customer: John
Pizza Type: Margherita
Quantity: 2
Total Price: $24
```

Problem 2: Weather Report Create a weather reporting system using console.log. **Sample Input:** City: "Mumbai", Temperature: 28°C, Humidity: 85%, Condition: "Rainy"
Problem 3: Gaming Score Display Display a player's gaming statistics. **Sample Input:** Player: "ProGamer123", Level: 45, Score: 89750, Lives: 3
Problem 4: Library Book Checkout Show book checkout information. **Sample Input:** Book: "Harry Potter", Author: "J.K. Rowling", Due Date: "2024-09-15"
Problem 5: Social Media Post Display a social media post with engagement stats. **Sample Input:** User: "@techguru", Post: "Learning JavaScript!", Likes: 127, Comments: 15, Shares: 8

2 Variables - The Storage Containers!

Variables are containers that store data values. In JavaScript, we have three ways to declare variables: `var`, `let`, and `const`.

2.1 Explanation

- **var**: Function-scoped, can be redeclared and updated (older way)
- **let**: Block-scoped, can be updated but not redeclared
- **const**: Block-scoped, cannot be updated or redeclared (constant)

2.2 Examples

Example 1: Variable Declarations

```javascript
// Different ways to declare variables
var oldWay = "I'm using var";
let modernWay = "I'm using let";
const constantValue = "I never change!";

console.log(oldWay);
console.log(modernWay);
console.log(constantValue);

// Updating variables
oldWay = "Updated var";
modernWay = "Updated let";
// constantValue = "Can't do this!"; // This would cause an
   error
```

Example 2: Scope Demonstration

```javascript
function demonstrateScope() {
    var varVariable = "I'm function scoped";

    if (true) {
        var varInBlock = "I'm still function scoped";
        let letInBlock = "I'm block scoped";
        const constInBlock = "I'm also block scoped";

        console.log(varInBlock);    // Works
        console.log(letInBlock);    // Works
        console.log(constInBlock);  // Works
    }

    console.log(varInBlock);    // Works - var is function
       scoped
    // console.log(letInBlock);    // Error - let is block
       scoped
    // console.log(constInBlock);  // Error - const is block
       scoped
}
```

2.3 Practice Questions

Practice Problems - Variables

Problem 1: Student Grade Calculator Create variables to store student information and calculate final grade.

```
// Use appropriate variable declarations
// Calculate final grade from assignments, midterm, and final
   exam
```

Sample Input: Assignments: 85, Midterm: 78, Final: 92, Weights: 40%, 30%, 30% **Expected Output:** Final Grade: 85.1
Problem 2: E-commerce Cart Manage shopping cart items with different variable types. **Sample Input:** Item: "Laptop", Price: 999.99, Quantity: 1, Tax Rate: 8.5%
Problem 3: Temperature Converter Convert temperatures between Celsius and Fahrenheit. **Sample Input:** Celsius: 25°C **Expected Output:** Fahrenheit: 77°F
Problem 4: Bank Account Manager Track account balance with deposits and withdrawals. **Sample Input:** Initial Balance: 1000, Deposit: 250, Withdrawal: 100
Problem 5: Recipe Scaler Scale recipe ingredients based on servings. **Sample Input:** Original servings: 4, Desired servings: 6, Flour: 2 cups
Problem 6: Fitness Tracker Calculate daily calorie burn from different activities. **Sample Input:** Running: 30 min (10 cal/min), Walking: 45 min (5 cal/min)
Problem 7: Movie Rating System Calculate average movie rating from multiple reviews. **Sample Input:** Reviews: [4.5, 3.8, 4.2, 4.7, 3.9]

3 If-Else Statements - Making Decisions!

Conditional statements allow your program to make decisions based on different conditions. They're like the brain of your program!

3.1 Explanation

`if-else` statements execute different code blocks based on whether a condition is true or false:

- **if**: Executes code if condition is true
- **else if**: Checks additional conditions
- **else**: Executes if all previous conditions are false

3.2 Examples

Example 1: Grade Classification

```javascript
let score = 85;
let grade;

if (score >= 90) {
    grade = "A";
    console.log("Excellent work!");
} else if (score >= 80) {
    grade = "B";
    console.log("Good job!");
} else if (score >= 70) {
    grade = "C";
    console.log("You can do better!");
} else if (score >= 60) {
    grade = "D";
    console.log("Study harder!");
} else {
    grade = "F";
    console.log("Please see me after class.");
}

console.log("Your grade is:", grade);
```

Output

```
Good job!
Your grade is: B
```

Example 2: Weather Clothing Advisor

```
let temperature = 15;
let isRaining = true;

if (temperature < 0) {
    console.log("Wear a heavy winter coat!");
} else if (temperature < 10) {
    console.log("Wear a warm jacket!");
} else if (temperature < 20) {
    console.log("Wear a light jacket or sweater!");
} else {
    console.log("T-shirt weather!");
}

if (isRaining) {
    console.log("Don't forget your umbrella!");
} else {
    console.log("No umbrella needed today!");
}
```

3.3 Practice Questions

Practice Problems - If-Else

Problem 1: ATM Withdrawal System Create an ATM system that checks balance before withdrawal. **Sample Input:** Balance: $500, Withdrawal Amount: $200 **Expected Output:** ”Withdrawal successful! New balance: $300”
Problem 2: Movie Ticket Pricing Calculate movie ticket prices based on age and day. **Sample Input:** Age: 65, Day: ”Tuesday” **Expected Output:** ”Senior discount + Tuesday special: $6”
Problem 3: Password Strength Checker Evaluate password strength based on length and characters. **Sample Input:** Password: ”MyP@ssw0rd123” **Expected Output:** ”Strong password!”
Problem 4: Traffic Light System Simulate a traffic light with appropriate actions. **Sample Input:** Light: ”Yellow” **Expected Output:** ”Prepare to stop!”
Problem 5: Restaurant Bill Calculator Calculate tip and total based on service quality. **Sample Input:** Bill: $50, Service: ”excellent” **Expected Output:** ”Tip: $10, Total: $60”
Problem 6: Gaming Character Stats Determine character class based on stats. **Sample Input:** Strength: 8, Intelligence: 15, Dexterity: 10 **Expected Output:** ”You are a Mage!”
Problem 7: Shipping Cost Calculator Calculate shipping based on weight and distance. **Sample Input:** Weight: 2.5kg, Distance: 150km **Expected Output:** ”Shipping cost: $12.50”
Problem 8: Exam Result Processor Process exam results with different outcomes. **Sample Input:** Score: 75, Attendance: 85% **Expected Output:** ”Passed with good attendance bonus!”

4 Loops - Repeat with Power!

Loops allow you to repeat code multiple times efficiently. JavaScript provides `for` and `while` loops for different scenarios.

4.1 Explanation

- **for loop**: Best when you know how many times to repeat
- **while loop**: Best when you repeat based on a condition
- **do-while loop**: Executes at least once, then checks condition

4.2 Examples

Example 1: For Loop - Countdown

```javascript
// Rocket launch countdown
console.log("Rocket Launch Countdown:");
for (let i = 10; i >= 1; i--) {
    console.log(i + "...");
}
console.log("BLAST OFF!");

// Sum of first 5 numbers
let sum = 0;
for (let i = 1; i <= 5; i++) {
    sum += i;
    console.log(`Adding ${i}, sum is now: ${sum}`);
}
console.log("Final sum:", sum);
```

Example 2: While Loop - Guessing Game

```javascript
// Simple number guessing simulation
let secretNumber = 7;
let guess = 1;
let attempts = 0;

while (guess !== secretNumber) {
    attempts++;
    console.log(`Attempt ${attempts}: Guessing ${guess}`);

    if (guess < secretNumber) {
        console.log("Too low!");
        guess++;
    } else if (guess > secretNumber) {
        console.log("Too high!");
        guess--;
    }
}

console.log("Correct! Found " + secretNumber + " in " +
   attempts + " attempts!");
```

4.3 Practice Questions

Practice Problems - Loops

Problem 1: Multiplication Table Generator Create a multiplication table for any number. **Sample Input:** Number: 7 **Expected Output:**

```
7 x 1 = 7
7 x 2 = 14
...
7 x 10 = 70
```

Problem 2: Star Pattern Printer Print different star patterns. **Sample Input:** Rows: 5 **Expected Output:** Triangle pattern with stars
Problem 3: Fibonacci Sequence Generator Generate first n Fibonacci numbers. **Sample Input:** n: 8 **Expected Output:** 0, 1, 1, 2, 3, 5, 8, 13
Problem 4: Prime Number Checker Check if numbers in a range are prime. **Sample Input:** Range: 10 to 20 **Expected Output:** Prime numbers: 11, 13, 17, 19
Problem 5: Shopping Cart Total Calculator Calculate total for multiple items with quantities. **Sample Input:** Items: [(price: 10, qty: 2), (price: 15, qty: 1), (price: 8, qty: 3)] **Expected Output:** Total: $59
Problem 6: Password Generator Generate random passwords of specified length. **Sample Input:** Length: 8 **Expected Output:** Random 8-character password
Problem 7: Grade Point Average Calculator Calculate GPA from multiple courses. **Sample Input:** Grades: [85, 92, 78, 88, 95] **Expected Output:** Average: 87.6
Problem 8: Digital Clock Simulator Simulate time progression. **Sample Input:** Start: 14:58, Duration: 5 minutes **Expected Output:** 14:58, 14:59, 15:00, 15:01, 15:02, 15:03

5 Break & Continue - Loop Control Masters!

`break` and `continue` statements give you fine control over loop execution, allowing you to exit early or skip iterations.

5.1 Explanation

- **break**: Immediately exits the loop completely
- **continue**: Skips the current iteration and moves to the next one

5.2 Examples

Example 1: Break - Finding First Even Number

```javascript
// Find the first even number in an array
let numbers = [1, 3, 5, 8, 9, 12, 15];
let firstEven = null;

for (let i = 0; i < numbers.length; i++) {
    console.log(`Checking number: ${numbers[i]}`);

    if (numbers[i] % 2 === 0) {
        firstEven = numbers[i];
        console.log(`Found first even number: ${firstEven}`);
        break; // Exit the loop immediately
    }
}

if (firstEven === null) {
    console.log("No even number found!");
}
```

Output

```
Checking number: 1
Checking number: 3
Checking number: 5
Checking number: 8
Found first even number: 8
```

Example 2: Continue - Processing Valid Scores

```
// Process only valid test scores (0-100)
let scores = [85, -5, 92, 150, 78, 88, -10, 95];
let validScores = [];
let totalValid = 0;

for (let i = 0; i < scores.length; i++) {
    console.log(`Processing score: ${scores[i]}`);

    // Skip invalid scores
    if (scores[i] < 0 || scores[i] > 100) {
        console.log("Invalid score " + scores[i] + " - 
           skipping");
        continue; // Skip to next iteration
    }

    console.log("Valid score: " + scores[i]);
    validScores.push(scores[i]);
    totalValid += scores[i];
}

console.log("Valid scores:", validScores);
console.log("Average of valid scores:", totalValid /
   validScores.length);
```

5.3 Practice Questions

Practice Problems - Break & Continue

Problem 1: Login Attempt Limiter Limit login attempts and break after successful login. **Sample Input:** Passwords: ["wrong1", "wrong2", "correct123"] **Expected Output:** "Login successful on attempt 3"
Problem 2: Prime Number Finder Find the first n prime numbers using break/continue. **Sample Input:** n: 5 **Expected Output:** [2, 3, 5, 7, 11]
Problem 3: Stock Price Alert System Monitor stock prices and break when target is reached. **Sample Input:** Prices: [100, 105, 98, 110, 115], Target: 110 **Expected Output:** "Target price $110 reached!"
Problem 4: Playlist Skip Controller Skip songs based on user preferences. **Sample Input:** Songs with ratings, Skip songs below rating 3 **Expected Output:** List of played songs
Problem 5: Game Level Progression Progress through game levels, skip bonus levels. **Sample Input:** Levels: [1, 2, "bonus1", 3, 4, "bonus2", 5] **Expected Output:** Completed main levels: [1, 2, 3, 4, 5]
Problem 6: Email Validator Validate email list, continue for invalid emails. **Sample Input:** Emails: ["user@email.com", "invalid-email", "test@test.com"] **Expected Output:** Valid emails processed count
Problem 7: Treasure Hunt Game Search for treasure, break when found. **Sample Input:** Grid positions, Treasure at position (3, 4) **Expected Output:** "Treasure found at position (3, 4) after 12 searches"
Problem 8: Chat Message Filter Filter chat messages, skip inappropriate content. **Sample Input:** Messages with content ratings **Expected Output:** Clean message list

6 Challenge Problems - Test Your Skills!

Advanced Challenge Problems

Challenge 1: Smart Calculator Create a calculator that processes multiple operations and handles errors. **Features:** Basic operations, error handling, memory functions **Sample Input:** Operations: ["5 + 3", "10 / 0", "7 * 6", "invalid"] **Expected Output:** Results with appropriate error messages
Challenge 2: Student Management System Build a complete student management system. **Features:** Add students, calculate grades, generate reports **Sample Data:** Multiple students with multiple subjects **Expected Output:** Class average, top performer, grade distribution
Challenge 3: Mini Banking System Implement a banking system with multiple account operations. **Features:** Deposits, withdrawals, transfers, balance inquiry, transaction history **Sample Input:** Various banking operations **Expected Output:** Account statements and balance updates
Challenge 4: Text Adventure Game Create a simple text-based adventure game. **Features:** Player movement, inventory, simple combat **Sample Input:** Player commands: ["north", "take sword", "attack goblin"] **Expected Output:** Game state updates and story progression
Challenge 5: Data Analytics Dashboard Process and analyze data from various sources. **Features:** Data filtering, calculations, trend analysis **Sample Input:** Sales data over multiple months **Expected Output:** Analytics summary with insights

7 Key Takeaways

Remember These Concepts

Console.log: Your debugging companion - use it to understand what your code is doing!
Variables: Choose the right type:

- Use `const` for values that won't change
- Use `let` for values that will change
- Avoid `var` in modern JavaScript

If-Else: Make your programs smart by adding decision-making logic!
Loops: Automate repetitive tasks efficiently:

- `for` when you know the number of iterations
- `while` when you loop based on conditions

Break & Continue: Fine-tune your loops for precise control!

Happy Coding!

Practice these concepts and you'll become a JavaScript pro!