
# JAVASCRIPT CONSOLE METHODS

SWIPE FOR MORE

# Introduction

The console object is part of the Web APIs provided by browsers (and also available in environments like Node.js). It gives developers powerful tools to inspect values, debug logic, measure performance, and understand runtime behavior.

While most developers only use console.log(), the console API is much deeper and more powerful.

## 1. console.log()

Purpose: General output logging.

```
console.log("Hello World");
console.log({ name: "Alex", age: 25 });
```

© Codeflare

### Key Notes

- Accepts multiple arguments
- Automatically formats objects
- Most commonly used method

## 2. console.info()

**Purpose**: Informational messages.

```
console.info("User successfully logged in");
```

### Notes

- Semantically similar to console.log()
- Some browsers display it with an info icon
- Useful for non-critical messages

© Codeflare

## 3. console.warn()

**Purpose**: Display warnings.

```
console.warn("This API is deprecated");
```

### Notes

- Displays a yellow warning icon
- Does not stop execution
- Ideal for highlighting risky behavior

## 4. console.error()

**Purpose**: Log errors.

```
console.error("Failed to fetch data");
```

© Codeflare

### Notes

- Displays a red error icon
- Includes stack trace in many browsers
- Useful for catching runtime issues

## 5. console.debug()

**Purpose**: Debug-level logging.

```
console.debug("Debugging value:", value);
```

### Notes

- Often hidden unless verbose logging is enabled
- Great for development-only logs

© Codeflare

# 6. console.table()

**Purpose**: Display tabular data.

```
console.table([
  { name: "Alice", score: 90 },
  { name: "Bob", score: 85 }
]);
```

## Notes

- Works best with arrays or objects
- Improves readability of structured data
- Supports column filtering

© Codeflare

# 7. console.group() & console.groupEnd()

**Purpose**: Group related logs.

```
console.group("User Details");
console.log("Name: Alex");
console.log("Age: 25");
console.groupEnd();
```

## Variants

- console.groupCollapsed() → starts collapsed

## Use Case

- Organizing complex logs
- Debugging nested processes

© Codeflare

## 8. console.time() & console.timeEnd()

**Purpose**: Measure execution time.

```
console.time("loop");
for (let i = 0; i < 1000000; i++) {}
console.timeEnd("loop");
```

### Notes

- Labels must match
- Great for performance testing

## 9. console.count()

**Purpose**: Count how many times something happens.

```
console.count("Click");
```

© Codeflare

**Notes**

- Label-based counter
- Useful in loops and event handlers

## 10. console.countReset()

**Purpose**: Reset a counter.

```
console.countReset("Click");
```

## 11. console.assert()

**Purpose**: Log errors only if a condition fails.

```
console.assert(age >= 18, "User is underage");
```

© Codeflare

### Notes

- Does nothing if condition is true
- Helpful for enforcing assumptions

## 12. console.clear()

**Purpose**: Clear the console.

```
console.clear();
```

### Notes

- Does not always fully clear in all browsers
- Useful during debugging sessions

© Codeflare