Complete JavaScript CheatSheet

- Developer Shaurya

Important Links

- **JavaScript Cheatsheet :** https://developershaurya.com/javascript-cheatsheet/
- **Website:** https://developershaurya.com/
- **All Cheatsheets :** https://developershaurya.com/cheat-sheets/
- **Youtube:** https://www.youtube.com/@DeveloperShaurya
- **(Developer Shaurya) Community :** https://t.me/developer_shaurya (Telegram)

JavaScript Cheatsheet – A Quick Guide for Beginners & Developers

by Developer Shaurya • September 2, 2025 • 0

JavaScript Complete Cheatsheet

Download Pdf

This Cheatsheet covers these all topic from basic to advance concept of JavaScript.

1. Basics

```
console.log("Hello, World!");   // Print
alert("Hello!");                // Alert popup
document.write("Hello JS");     // Write on page
```

2. Variables

```
var a = 10;    // Function-scoped
let b = 20;    // Block-scoped
const c = 30;  // Constant
```

3. Data Types

```
// Primitive
let str = "Hello";     // String
let num = 42;          // Number
let bool = true;       // Boolean
let undef;             // Undefined
let empty = null;      // Null
let big = 123n;        // BigInt
let sym = Symbol("id");// Symbol

// Reference
let arr = [1, 2, 3];
let obj = {name: "Ravi"};
```

4. Operators

```
// Arithmetic
+ - * / % **

// Assignment
= += -= *= /= %=

// Comparison
== === != !== > < >= <=

// Logical
&& || !

// Ternary
let result = age >= 18 ? "Adult" : "Minor";
```

5. Strings

```
let text = "JavaScript";
text.length;              // 10
text.toUpperCase();       // "JAVASCRIPT"
text.includes("Script");  // true
text.slice(0, 4);         // "Java"

// Template literals
let name = "Ravi";
`Hello, ${name}`;
```

6. Numbers & Math

```
Math.round(4.6);   // 5
Math.floor(4.6);   // 4
Math.ceil(4.1);    // 5
Math.random();     // 0-1 random
Math.max(1,5,9);   // 9
```

7. Arrays

```
let fruits = ["apple", "banana"];
fruits.push("mango");   // Add end
fruits.pop();           // Remove end
fruits.shift();         // Remove start
fruits.unshift("kiwi"); // Add start

// Higher-order methods
fruits.map(f => f.toUpperCase());
fruits.filter(f => f.startsWith("a"));
fruits.reduce((a,b) => a+b, 0);
```

8. Objects

```
let person = {
  name: "Ravi",
  age: 20,
  greet() { console.log("Hello"); }
};
console.log(person.name);
person.greet();
```

9. Functions

```
function add(a,b) { return a+b; }
const multiply = (a,b) => a*b;
(function(){ console.log("IIFE"); })();
```

10. Control Flow

```
if (x > 10) { ... }
else if (x === 10) { ... }
else { ... }

switch(day) {
  case "Mon": console.log("Start"); break;
  default: console.log("Other");
}

for (let i=0; i<5; i++) console.log(i);
while(x < 5) { x++; }
do { x++; } while(x < 5);
```

11. DOM Manipulation

```
document.getElementById("id");
document.querySelector(".class");

element.innerHTML = "New Text";
element.style.color = "red";

let p = document.createElement("p");
p.textContent = "Hello";
document.body.appendChild(p);
```

12. Events

```
document.getElementById("btn").addEventListener("click", () => {
  alert("Clicked!");
});
```

13. ES6+ Features

```
// Destructuring
let [a,b] = [1,2];
let {name, age} = {name:"Ravi", age:20};

// Spread & Rest
let arr1 = [1,2];
let arr2 = [...arr1, 3,4];
function sum(...nums){ return nums.reduce((a,b)=>a+b); }

// Default params
function greet(name="Guest"){ return `Hello, ${name}`; }
```

14. Classes & OOP

```
class Person {
  constructor(name){ this.name = name; }
  greet(){ console.log(`Hello ${this.name}`); }
}
class Student extends Person {
  constructor(name, roll){ super(name); this.roll = roll; }
}
```

15. Asynchronous JS

```
// Callbacks
setTimeout(() => console.log("Done"), 1000);

// Promises
let p = new Promise((res,rej)=> res("Success"));
p.then(res => console.log(res));

// Async/Await
async function fetchData(){
  let res = await fetch("https://jsonplaceholder.typicode.com/posts/1");
  let data = await res.json();
  console.log(data);
}
```

16. Advanced Built-ins

```
// Map & Set
let map = new Map([["a",1]]);
map.set("b",2);
map.get("a");

let set = new Set([1,2,2,3]); // {1,2,3}

// WeakMap & WeakSet (store weak refs)
```

17. Error Handling

```
try {
  throw new Error("Something went wrong");
} catch(e) {
  console.log(e.message);
} finally {
  console.log("Always runs");
}
```

18. Regular Expressions

```
let regex = /hello/i;
regex.test("Hello world");   // true
"Hello".match(/h/i);         // ["H"]
"123".replace(/\d/g, "#");   // "###"
```

19. Modules

```
// export.js
export const pi = 3.14;
export default function greet(){ console.log("Hi"); }

// import.js
import greet, {pi} from "./export.js";
```

20. Browser APIs

```
// Local Storage
localStorage.setItem("name", "Ravi");
localStorage.getItem("name");

// Geolocation
navigator.geolocation.getCurrentPosition(pos => console.log(pos));

// Alert & Confirm
alert("Hello");
confirm("Are you sure?");
prompt("Enter name:");
```

21. Modern Features

```
// Optional chaining
let city = user?.address?.city;

// Nullish coalescing
let value = input ?? "default";

// Private class fields
class A {
  #secret = 123;
  getSecret(){ return this.#secret; }
}
```

22. Best Practices

- Use `let` & `const`, avoid `var`.
- Always use `===` instead of `==`.
- Keep functions small & reusable.
- Avoid polluting global scope.
- Use async/await for cleaner async code.

Conclusion

This **JavaScript Cheatsheet** includes everything: from **basics → advanced concepts → ES6+ features → browser APIs**. Keep it bookmarked as your **ultimate quick reference** while coding.