

# ■ JavaScript Cheat Sheet

## 1. Variables

```
var a = 10;    // function-scoped
let b = 20;    // block-scoped
const c = 30; // block-scoped constant
```

## 2. Data Types

String	→ "hello"
Number	→ 42
Boolean	→ true / false
Null	→ null
Undefined	→ undefined
Object	→ { key: "value" }
Array	→ [1, 2, 3]
Symbol	→ unique value
BigInt	→ 9007199254740991n

## 3. Functions

```
// Function Declaration
function greet(name) {
  return "Hello " + name;
}

// Function Expression
const add = function(a, b) { return a + b; };

// Arrow Function
const multiply = (a, b) => a * b;
```

## 4. Conditionals

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

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

## 5. Loops

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

let arr = [1,2,3];
for (let val of arr) { console.log(val); }

arr.forEach(v => console.log(v));
```

```
while(condition) { ... }
do { ... } while(condition);
```

## 6. Objects

```
let user = { name: "Alex", age: 25 };
console.log(user.name);      // dot notation
console.log(user["age"]);    // bracket notation
```

## 7. ES6+ Features

```
// Template literals
let name = "Alex";
console.log(`Hello ${name}`);

// Destructuring
let { age } = user;
let [x, y] = [10, 20];

// Spread & Rest
let arr2 = [...arr, 4, 5];
function sum(...nums) { return nums.reduce((a,b)=>a+b); }

// Default params
function sum(a=1, b=2){ return a+b; }
```

## 8. Optional Chaining

```
let user = { profile: { name: "Alex" } };
console.log(user?.profile?.name); // "Alex"
console.log(user?.address?.city); // undefined (safe!)
```

## 9. Hoisting

```
console.log(a); // undefined
var a = 5;

sayHi(); // works
function sayHi() { console.log("Hi!"); }

console.log(b); // ReferenceError
let b = 10;
```

## 10. Closures

```
function outer() {
  let count = 0;
  return function inner() {
    count++;
    return count;
  };
}
let counter = outer();
```

```
console.log(counter()); // 1
console.log(counter()); // 2
```

## 11. Promises & Async/Await

```
// Promise
let promise = new Promise((resolve, reject) => {
  setTimeout(() => resolve("Done!"), 1000);
});
promise.then(result => console.log(result));

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

## 12. Common Array Methods

```
let arr = [1,2,3,4,5];

arr.map(x => x*2);          // [2,4,6,8,10]
arr.filter(x => x%2==0);    // [2,4]
arr.reduce((a,b)=>a+b);    // 15
arr.find(x => x>3);       // 4
arr.includes(3);            // true
```