JavaScript Interview — 50 High■Yield Q&A; (with quick answers & examples)

This document provides 50 JavaScript interview-style questions with concise answers and code snippets/examples.

Q1. Difference between var, let, and const?

Quick answer: var: function-scoped, hoisted. let: block-scoped, no redeclare. const: block-scoped, constant reference.

Example: let x=5; const y=10; var z=20;

Q2. What are arrow functions?

Quick answer: Shorter syntax, no own this/arguments, lexical scoping.

Example: const add=(a,b)=>a+b;

Q3. Difference between == and ===?

Quick answer: == checks value with coercion. === checks value and type (strict).

Example: 0=='0' //true, 0==='0' //false

Q4. Explain closures.

Quick answer: Function remembers variables from its lexical scope even after scope ends.

Example: function outer(){let x=10; return ()=>x;}

Q5. What is hoisting?

Quick answer: Declarations are moved to top. var undefined; functions fully hoisted.

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

Q6. Explain event bubbling and capturing.

Quick answer: Bubbling: child to parent. Capturing: parent to child. Controlled by addEventListener third param.

Example: elem.addEventListener('click',fn,true); //capturing

Q7. What are promises?

Quick answer: Objects representing eventual success/failure of async operations.

Example: new Promise((res,rej)=>res(42)).then(v=>console.log(v));

Q8. Explain async/await.

Quick answer: Syntactic sugar over promises; await pauses until resolved.

Example: async function f(){ let d=await fetch('/'); }

Q9. Difference between null and undefined?

Quick answer: undefined: not assigned. null: explicitly empty.

Example: let a; console.log(a); //undefined a=null;

Q10. Explain prototype in JS.

Quick answer: Objects inherit properties via prototype chain.

Example: function Person(){} Person.prototype.sayHi=function(){};

Q11. Difference between function declaration and expression?

Quick answer: Declaration hoisted; expression not hoisted.

Example: function f(){} const g=function(){};

Q12. What is DOM?

Quick answer: Document Object Model, tree representation of HTML.

Example: document.getElementById('id');

Q13. Difference between map and forEach?

Quick answer: map returns new array, forEach does not return.

Example: [1,2].map(x=>x*2);

Q14. What is event delegation?

Quick answer: Use parent listener to handle child events via bubbling.

Example: ul.addEventListener('click',e=>{if(e.target.tagName==='LI'){…}})

Q15. Difference between call, apply, and bind?

Quick answer: call: invoke with args list. apply: invoke with args array. bind: returns new function with bound this.

Example: fn.call(obj,1,2); fn.apply(obj,[1,2]); const g=fn.bind(obj);

Q16. What is NaN?

Quick answer: Not a Number, type Number, result of invalid math.

Example: typeof NaN //number

Q17. Explain strict mode.

Quick answer: Adds stricter parsing, eliminates silent errors.

Example: 'use strict'; var x=010; //error

Q18. What are template literals?

Quick answer: String with backticks, supports interpolation and multiline.

Example: `Hello ${name}`

Q19. Difference between slice and splice?

Quick answer: slice: returns new array portion. splice: modifies array in place.

Example: arr.slice(1,3); arr.splice(1,2);

Q20. Explain setTimeout and setInterval.

Quick answer: setTimeout: once after delay. setInterval: repeatedly after delay.

Example: setTimeout(()=>console.log('hi'),1000);

Q21. What is JSON?

Quick answer: JavaScript Object Notation, lightweight data format.

Example: JSON.parse('{"x":1}'); JSON.stringify({x:1});

Q22. Difference between undefined and not defined?

Quick answer: undefined: declared but no value. not defined: variable never declared.

Example: console.log(x) //ReferenceError

Q23. What is NaN === NaN?

Quick answer: Always false; NaN is not equal to anything including itself.

Example: NaN===NaN //false

Q24. Explain ES6 modules.

Quick answer: import/export syntax for modular JS.

Example: export default f; import f from './f.js';

Q25. Difference between shallow copy and deep copy?

Quick answer: Shallow: copies refs. Deep: copies nested objects.

Example: let a={x:{y:1}}; let b={...a};

Q26. What is an IIFE?

Quick answer: Immediately Invoked Function Expression, runs immediately.

Example: (function(){console.log('run')})();

Q27. What is event loop?

Quick answer: Handles async callbacks; tasks queued in micro/macrotasks.

Example: Promise.resolve().then(()=>{}); setTimeout(()=>{});

Q28. Difference between synchronous and asynchronous code?

Quick answer: Sync: executes line by line. Async: callbacks/promises/event loop.

Example: fs.readFile vs fs.readFileSync (Node.js).

Q29. What are generators?

Quick answer: Functions that yield multiple values using function* and yield.

Example: function* g(){yield 1; yield 2;}

Q30. What is typeof null?

Quick answer: Returns 'object' (bug in JS, legacy).

Example: typeof null //'object'

Q31. Explain destructuring assignment.

Quick answer: Extract values from arrays/objects into variables.

Example: const {a,b}=obj; const [x,y]=arr;

Q32. What is spread operator?

Quick answer: Expands iterable elements into places.

Example: let arr=[1,2]; let b=[...arr,3];

Q33. Difference between for..in and for..of?

Quick answer: for..in: keys of object. for..of: values of iterable.

Example: for(let i in obj){} for(let v of arr){}

Q34. What are symbols?

Quick answer: Unique and immutable primitive keys.

Example: let s=Symbol('id');

Q35. Difference between localStorage, sessionStorage, and cookies?

Quick answer: localStorage: persistent. sessionStorage: per tab. cookies: sent to server, smaller size.

Example: localStorage.setItem('k','v');

Q36. What is debouncing?

Quick answer: Delays execution until no more events within timeframe.

Example: function debounce(fn,d){…}

Q37. What is throttling?

Quick answer: Ensures function runs at most once per timeframe.

Example: function throttle(fn,d){…}

Q38. What is NaN property of Number?

Quick answer: Number.NaN is NaN constant.

Example: Number.isNaN(NaN) //true

Q39. Difference between mutable and immutable objects?

Quick answer: Mutable can be changed (arrays). Immutable not directly changed (strings).

Example: let s='a'; s[0]='b'; //no change

Q40. What is difference between var hoisting and let/const hoisting?

Quick answer: var hoisted undefined. let/const hoisted in temporal dead zone.

Example: console.log(x); let x=5; //ReferenceError

Q41. Explain shallow vs deep equality?

Quick answer: === compares by value for primitives, by reference for objects.

Example: {}==={} //false

Q42. What are WeakMap and WeakSet?

Quick answer: Collections with weak refs to objects, no memory leaks, keys must be objects.

Example: let wm=new WeakMap();

Q43. What is difference between Array.map and Array.filter?

Quick answer: map transforms each element. filter keeps matching elements.

Example: [1,2,3].filter(x=>x>1);

Q44. Explain Promise.all.

Quick answer: Runs promises in parallel; resolves when all resolved or rejects on one failure.

Example: Promise.all([p1,p2]).then(res=>{});

Q45. What are async iterators?

Quick answer: Iterators returning promises, used with for-await-of.

Example: for await (let v of asyncGen()){}

Q46. What is optional chaining?

Quick answer: ?. allows safe access to nested properties.

Example: let city=user?.address?.city;

Q47. Explain nullish coalescing operator.

Quick answer: ?? returns right side only if left is null/undefined.

Example: let x=a??'default';

Q48. Difference between shallow comparison and deep comparison?

Quick answer: Shallow checks references. Deep checks nested values.

Example: JSON.stringify(a)===JSON.stringify(b);

Q49. Additional JavaScript concept #49

Quick answer: Explanation of a key JavaScript interview topic.

Example: Example usage or snippet here.

Q50. Additional JavaScript concept #50

Quick answer: Explanation of a key JavaScript interview topic.

Example: Example usage or snippet here.