Sheryians Presents

JavaScript Interview **Essentials**

1. What is hoisting?

JavaScript moves function and variable declarations to the top of their scope before running the code.

👉 That's why you can use a function before it's written.

2. Difference between == and ===

- == checks value only (does type conversion).
- === checks value + type (strict comparison).

👉 Always use === for accuracy.

3. What is a closure? Use case?

A function that remembers the variables from its outer scope even after the outer function has finished.

👉 Used in private variables, currying, and factory functions.

4. What is Event Delegation?

Instead of adding listeners to many child elements, add one to the parent and handle events using event.target.
👉 Improves performance and handles dynamic elements.

5. Shallow vs Deep Copy

- Shallow Copy → Only top-level is copied; nested objects still linked.
- Deep Copy → Full independent clone, including nested values.

6. Call, Apply, Bind

All three set this manually.

- call() → runs function with args
- apply() → same as call but with array
- bind() → returns new function with this fixed

7. this in arrow vs normal functions

- Arrow functions don't have their own this → they use the parent's.
- Normal functions get this based on how they're called.

8. Prototype and Inheritance

All JS objects inherit from a prototype.
You can share methods across objects using prototype chaining.

9. Difference: map, reduce, filter

- map() → transforms every item → returns new array
- filter() → removes items → returns new array
- reduce() → turns array into a single value

10. How does async/await work internally?

It's just a prettier way to write Promises.
Behind the scenes, it's still using .then() and a Promise chain.

11. Difference between null and undefined

- undefined → variable declared but not assigned
- null → intentional empty value assigned by you

12. Debounce vs Throttle

- Debounce → runs a function after pause in activity
- Throttle → runs a function every X ms, no matter how often event fires

13. Memory Leaks in JS

When something (like a timer or event listener) keeps using memory even when it's no longer needed → app gets slow over time.

14. Event Loop Phases

Controls how JS handles tasks.

- Call stack runs sync code
- Web APIs handle async (like setTimeout)
- Event loop moves ready callbacks to stack when it's empty