JavaScript OOP Interview Guide

1. Classes and Objects

Q: Q: What are classes and objects in JavaScript?

A: A class is a blueprint for creating objects. An object is an instance of a class.

Example:

class Person { constructor(name, age) { this.name = name; this.age = age; } } const user = new Person('Jugal', 25);

2. Constructor

Q: Q: What is the role of the constructor method in a class?

A: It initializes object properties when a new object is created.

Example:

class Car { constructor(brand) { this.brand = brand; } } const c = new Car('BMW');

3. 'this' Keyword

Q: Q: How does the 'this' keyword behave in classes?

A: Inside class methods, 'this' refers to the current instance of the class.

Example:

class User { constructor(name) { this.name = name; } show() { console.log(this.name); } }

4. Inheritance

Q: Q: How is inheritance implemented in JavaScript?

A: Using the 'extends' keyword. Child classes can use 'super()' to call the parent constructor.

Example:

class Animal { speak() { console.log('Animal speaks'); } } class Dog extends Animal { speak() { console.log('Dog barks'); } }

5. Encapsulation

Q: Q: How do you achieve encapsulation in JavaScript?

A: By using private fields (#) or closure-based data hiding.

Example:

class Account { #balance = 0; deposit(amount) { this.#balance += amount; } getBalance() { return this.#balance; } }

6. Polymorphism

Q: Q: What is polymorphism in OOP?

A: It allows the same method to behave differently depending on the object.

Example:

class Shape { draw() { console.log('Drawing shape'); } } class Circle extends Shape { draw() { console.log('Drawing circle'); } }

7. Abstraction

Q: Q: How can abstraction be implemented in JavaScript?

A: By creating base classes with methods meant to be overridden by subclasses.

Example:

class Payment { pay() { throw new Error('Implement this method'); } } class UpiPayment extends Payment { pay() { console.log('Paying via UPI'); } }

8. Static Methods and Properties

Q: Q: What are static methods and when should they be used?

A: Static methods belong to the class, not instances. Useful for utility or shared logic.

Example:

class MathUtil { static add(a, b) { return a + b; } } console.log(MathUtil.add(2, 3));

9. Prototype & Inheritance Chain

Q: Q: What is the prototype chain?

A: It's the mechanism by which objects inherit features from other objects.

Example:

function Person(name) { this.name = name; } Person.prototype.sayHi = function() { console.log('Hi ' + this.name); };

10. Getters and Setters

Q: Q: What are getters and setters used for?

A: They provide controlled access to object properties.

Example:

class User { constructor(name) { this._name = name; } get name() { return this._name.toUpperCase(); } set name(val) { this._name = val.trim(); } }

End of JavaScript OOP Interview Guide