**Armen Melkumyan** • 1st

Technical / Solutions Architect

1yr • 🌐

• • •

From interview questions.
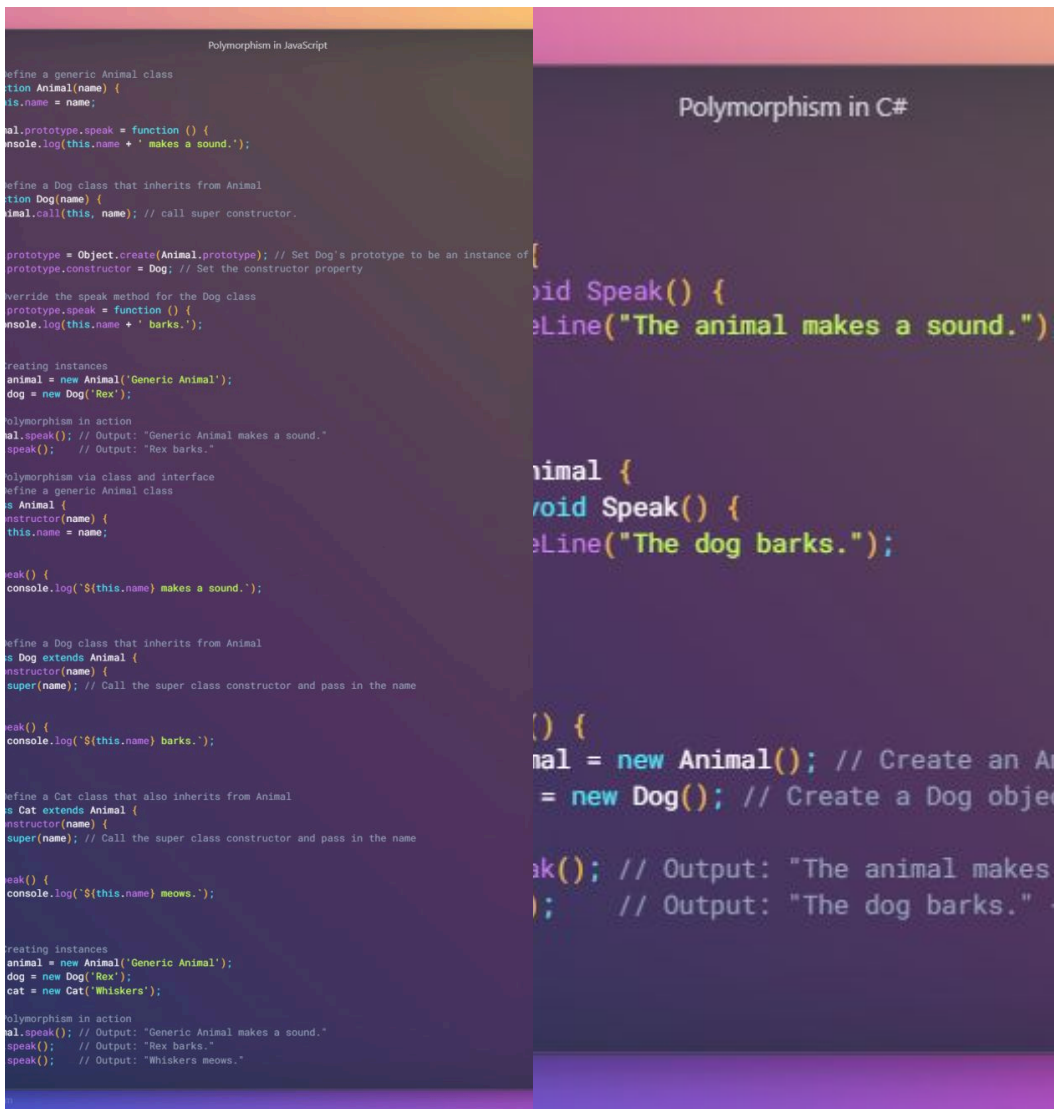
What is Polymorphism ?

Polymorphism in Object-Oriented Programming (OOP) is a concept that refers to the ability of different objects to respond in different ways to the same message, function, or operation. It's one of the four fundamental OOP concepts, alongside encapsulation, inheritance, and abstraction. Polymorphism allows objects of different classes to be treated as objects of a common superclass.

Polymorphism in JavaScript

In JavaScript, polymorphism can be achieved through prototypes and inheritance and can be implemented using ES6 classes, which offer a more straightforward syntax compared to the prototype-based approach.

In C#, polymorphism is usually achieved through inheritance and the use of virtual methods or interfaces.

#Polimophism #OOP #javascript #charp #interviewquestions #interviewpreparation #prototypes #class #interface

```javascript
efine a generic Animal class
tion Animal(name) {
is.name = name;

al.prototype.speak = function () {
nsole.log(this.name + ' makes a sound.');


efine a Dog class that inherits from Animal
tion Dog(name) {
imal.call(this, name); // call super constructor.


prototype = Object.create(Animal.prototype); // Set Dog's prototype to be an instance of
prototype.constructor = Dog; // Set the constructor property

verride the speak method for the Dog class
prototype.speak = function () {
nsole.log(this.name + ' barks.');


reating instances
animal = new Animal('Generic Animal');
dog = new Dog('Rex');

olymorphism in action
al.speak(); // Output: "Generic Animal makes a sound."
speak();    // Output: "Rex barks."

olymorphism via class and interface
efine a generic Animal class
s Animal {
nstructor(name) {
this.name = name;


eak() {
console.log(`${this.name} makes a sound.`);


efine a Dog class that inherits from Animal
s Dog extends Animal {
nstructor(name) {
super(name); // Call the super class constructor and pass in the name


eak() {
console.log(`${this.name} barks.`);


efine a Cat class that also inherits from Animal
s Cat extends Animal {
nstructor(name) {
super(name); // Call the super class constructor and pass in the name


eak() {
console.log(`${this.name} meows.`);


reating instances
animal = new Animal('Generic Animal');
dog = new Dog('Rex');
cat = new Cat('Whiskers');

olymorphism in action
al.speak(); // Output: "Generic Animal makes a sound."
speak();    // Output: "Rex barks."
speak();    // Output: "Whiskers meows."
```

## Polymorphism in C#

```csharp
oid Speak() {
eLine("The animal makes a sound.");


imal {
void Speak() {
eLine("The dog barks.");


) {
al = new Animal(); // Create an An
= new Dog(); // Create a Dog objec

k(); // Output: "The animal makes
);    // Output: "The dog barks." —
```