upGrad ENTERPRISE

Object-Oriented Concepts in C#: Polymorphism

By
Narasimha Rao T
Microsoft.Net FSD Trainer
Professional Development Trainer
tnrao.trainer@gmail.com

1. Polymorphism

Definition:

Polymorphism allows objects to be treated as instances of their parent class rather than their actual class. It enables the same method to behave differently based on the object instance.

Types:

- **Compile-time polymorphism (Static Binding):** Achieved via method overloading and operator overloading.
- **Run-time polymorphism (Dynamic Binding):** Achieved via method overriding using inheritance and `virtual` / `override` keywords.

2. Example:

```
class Animal {
    public virtual void Speak() {
        Console.WriteLine("Animal speaks");
    }
}

class Dog : Animal {
    public override void Speak() {
        Console.WriteLine("Dog barks");
    }
}
```

3. Method Overriding

Definition:

Overriding allows a subclass to provide a specific implementation of a method already defined in its base class.

Keywords:

- `virtual` → in base class
- `override` → in derived class

Example:

```
class Parent {
    public virtual void Print() {
        Console.WriteLine("Parent Print");
    }
}

class Child : Parent {
    public override void Print() {
        Console.WriteLine("Child Print");
    }
}
```

4. Method Overloading

Definition:

Method overloading allows multiple methods in the same class with the same name but different **parameter lists** (type, number, or order).

Example:

```
class MathOps {
    public int Add(int a, int b) {
        return a + b;
    }

    public double Add(double a, double b) {
        return a + b;
    }
}
```

Key Points:

- No implementation in the interface itself.
- A class can implement **multiple interfaces** (unlike class inheritance).
- Members are public by default; access modifiers are not allowed.

upGrad ENTERPRISE

Q & A

Narasimha Rao T
tnrao.trainer@gmail.com