Polymorphism– Week 6

Written by: Sergio Luilly Cabrera Dorado

1. Significado de Polimorfismo

Poliformismo significa “muchas formas”, en POO es cuando un mismo código o programa o proyecto, puede tener comportamientos diferentes dependiendo de su contexto.

Es un principio de OOP (programación orientada a objetos) que permite que un mismo método o función se comporte de manera diferente según los diferentes tipos de objetos con los que se este quiera trabajar. Se podría asumir que es similar a una herramienta que puede tomar diversas funcionalidades y que sabe adaptarse a diferentes situaciones.

1. Beneficio Principal

La mayor ventaja del polimorfismo es la **flexibilidad y extensibilidad del código**. Permite que se puedan agregar nuevos tipos de objetos, sin modificar el código existente que los utiliza o crea, haciendo que el programa sea más mantenible y escalable, además integrar todos los demás principios de la OOP

1. Aplicación Práctica

En nuestro programa Eternal Quest, aplicamos polimorfismo para manejar diferentes tipos de metas (Simple, Eternal, Checklist) que comparten la misma interfaz pero implementan comportamientos específicos para calcular puntos y mostrar información.

1. Código

// POLIMORFISMO EN ACCIÓN - misma línea, diferentes comportamientos

foreach (Goal goal in \_goals)

{

// Esta misma línea llama a métodos diferentes según el tipo de meta

string details = goal.GetDetailsString();

Console.WriteLine(details);

}

En este bucle, la línea goal.GetDetailsString() es polimórfica porque:

* Si goal es un **SimpleGoal**, muestra: [X] Run Marathon (Complete marathon)
* Si goal es un **EternalGoal**, muestra: [ ] Read Scriptures - Completed 5 times
* Si goal es un **ChecklistGoal**, muestra: [ ] Attend Temple - Completed 3/10 times

**La misma instrucción produce resultados diferentes** porque cada clase override el método GetDetailsString() con su propia implementación."

*De la supeclase*

public abstract class Goal

{

public abstract string GetDetailsString(); // Método abstracto

}

*De las clases hijas*

// SimpleGoal

public override string GetDetailsString()

{

return $"{(\_isComplete ? "[X]" : "[ ]")} {\_name} ({\_description})";

}

// EternalGoal

public override string GetDetailsString()

{

return $"[ ] {\_name} ({\_description}) - Completed {\_timesCompleted} times";

}

// ChecklistGoal

public override string GetDetailsString()

{

return $"{(\_amountCompleted >= \_target ? "[X]" : "[ ]")} {\_name} - Completed {\_amountCompleted}/{\_target}";

}