www.PragimTech.com

Training +Placements = Our success

Pragim@PragimTech.com

Parte 26 – ¿Por qué usar propiedades en C#?

Cristina Carrasco

PRAGIM Technologies

Cristina@cristinacarrasco.com

En esta lección veremos:

- La necesidad de las propiedades

¿Por qué usar propiedades?

Marcar los campos de una clase como públicos y exponerlos afuera de la clase es una mala practica, porque perdemos el control sobre que valores pueden tomar y retornar estos campos.

```
public class Estudiante
{
    public int Id;
    public string Nombre;
    public int CalificacionAprobatoria = 60;
}
public class Program
{
    static void Main(string[] args)
    {
      Estudiante est = new Estudiante();
      est.Id = -101;
      est.Nombre = null;
      est.CalificacionApprobatoria = 0;
      Console.WriteLine($"Id={est.Id}, Nombre={est.Nombre},Cal.Aprob={est.CalificacionApprobatoria}");
    }
}
```

Problema con los campos públicos

1. Id no debe ser negativo
2. El nombre no puede ser nulo
3. Si el nombre no fue definido se debe retornar “Sin nombre”
4. La calificación aprobatoria debe ser de solo lectura.

Métodos Getter y Setter

Los lenguajes de programación que no tienen propiedades utilizan los métodos getter y setter para encapsular y proteger campos.

```
public class Estudiante
{
    private int _id;
    public string _nombre;
    public int _calificacionAprobatoria;
    public void SetId(int id)
    {
        if(id <= 0)
        {
            throw new Exception("El Id del estudiante debe ser mayor que cero");
        }
        this._id = id;
    }
    public int GetId()
    {
        return _id;
    }
}
public class Program
{
    static void Main(string[] args)
    {
        Estudiante est = new Estudiante();
        est.SetId(101);
        Console.WriteLine($"Id del estudiante={est.GetId()}");
    }
}
```

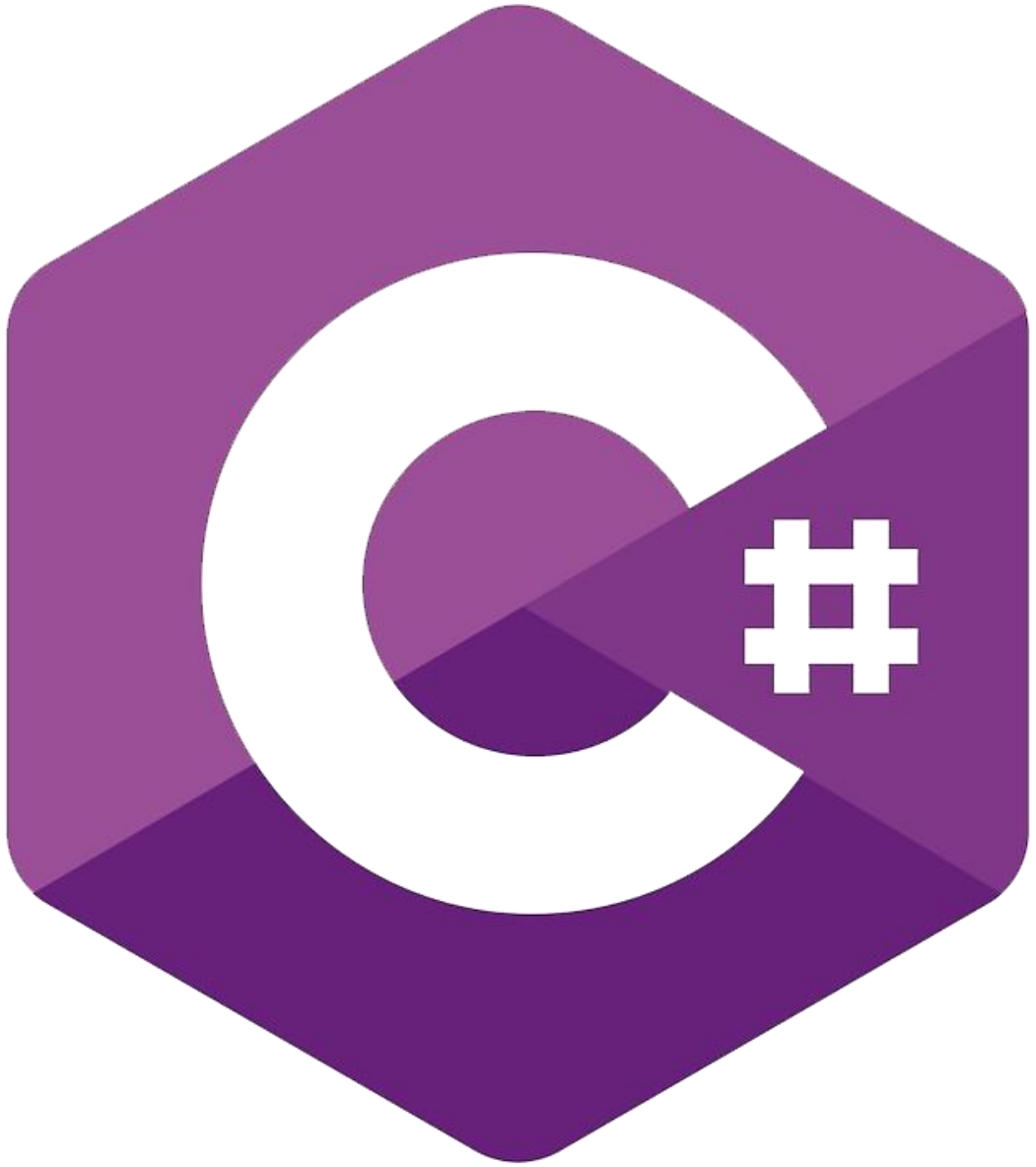
En este ejemplo utilizamos los métodos SetId(int id) y GetId() para encapsular el campo _id.

Como resultado témenos mejor control sobre lo que asignamos y retornamos para el campo _id.

Nota: El encapsulamiento es uno de los pilares principales de la programación orientada a objetos.

Recursos adicionales

- PRAGIM Pagina principal
  - http://www.PragimTech.com



[26]

¿Por qué usar Propiedades?