ULBRA - TORRES

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

AP2 - POO LIVRARIA

Miguel Elias Bernardes

Lucas Rodrigues Schwartzhaupt Fogaça

Torres - RS

2024

INTRODUÇÃO

O trabalho a seguir irá explicar os principais pilares do paradigma de POO(Programação Orientada a Objetos) que tem como principal objetivo simplificar e agilizar a correção e/ou alteração das linhas de código, evitando a repetição e desorganização do mesmo!

Após as explicações terão exemplos de código real com esses pilares. O código será de uma livraria que terá as principais funções:

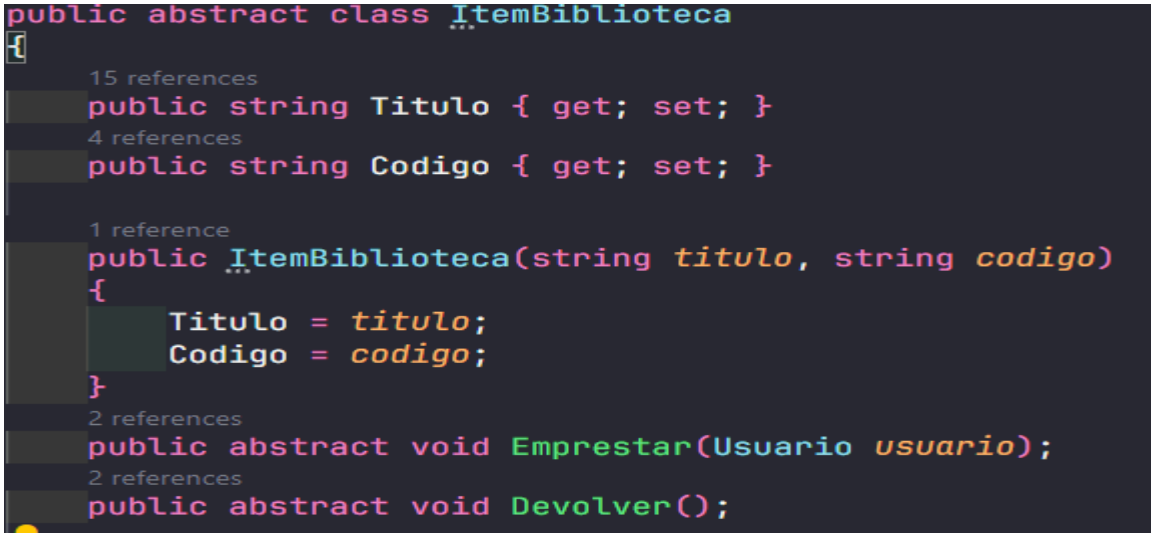
- Cadastro de livros e usuários
- Alterar dados dos cadastros
- Emprestar livros
- Devolver livros.

DESENVOLVIMENTO

A Programação Orientada a Objetos tem como seus principais quatros pilares:

- **ENCAPSULAMENTO -** Esse pilar é para a proteção na alteração de dados, para que você possa alterar só os dados permitidos pelo meio de getters e setters, para que não haja alguma alteração nos dados privados o que geraria problemas no funcionamento do código como um todo.

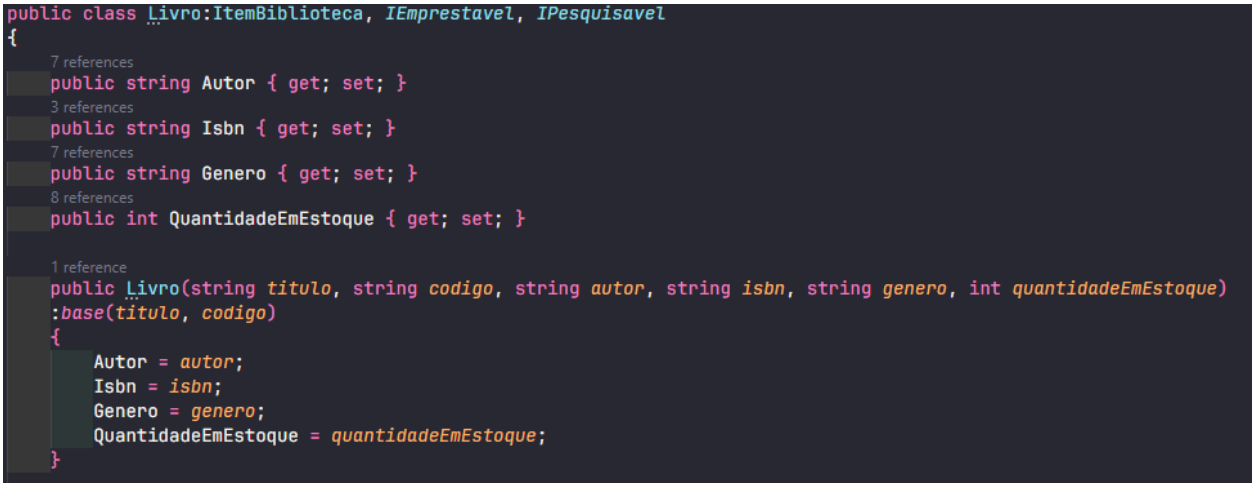
  Exemplo:



No exemplo acima estamos criando os métodos Emprestar e Devolver como abstratos dentro da classe ItemBiblioteca, esses métodos forçam as classes derivantes a terem que criar seus próprios métodos. Assim a implementação real de Emprestar e Devolver está encapsulada dentro das classes que herdam de ItemBiblioteca.

- **HERANÇA -** A herança serve para pegar dados de uma classe pai, digamos que temos a classe pai e criamos a classe filho herdando a classe pai. A classe filho terá todos os dados da classe pai e mais os seus próprios dados.

  Exemplo:



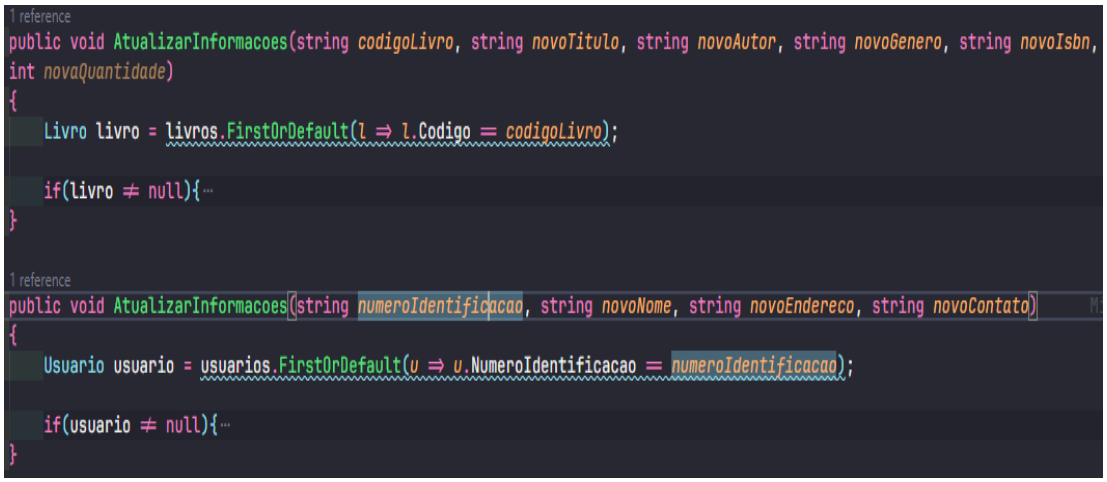
```
public class Livro:ItemBiblioteca, IEmprestavel, IPesquisavel
{
    7 references
    public string Autor { get; set; }
    3 references
    public string Isbn { get; set; }
    7 references
    public string Genero { get; set; }
    8 references
    public int QuantidadeEmEstoque { get; set; }

    1 reference
    public Livro(string titulo, string codigo, string autor, string isbn, string genero, int quantidadeEmEstoque)
    :base(titulo, codigo)
    {
        Autor = autor;
        Isbn = isbn;
        Genero = genero;
        QuantidadeEmEstoque = quantidadeEmEstoque;
    }
```

No exemplo acima temos a classe Livro que está herdando a classe ItemBiblioteca e com isso está recebendo os atributos Titulo e Codigo da classe pai e tem os seus próprios atributos como Autor, Isbn, Genero, QuantidadeEmEstoque.

- **POLIMORFISMO -** O polimorfismo é quando você utiliza um mesmo método para fazer mais de uma função, por exemplo um método que altera os dados de um cliente ou livro, pode ser o mesmo método mas recebendo propriedades diferentes e assim fazendo a função definida para cada tipo de item que está sendo atualizado, esse processo também é conhecido por sobrecarga de método.

  Exemplo:



No exemplo acima, utilizamos um método com o mesmo nome AtualizarInformações para dois itens diferentes, então só colocamos propriedades diferentes, assim fazendo uma sobrecarga.

- **ABSTRAÇÃO -** A abstração é você criar um objeto do mundo real no código porém sem colocar características específicas que não são necessárias, por exemplo um carro, ele tem os métodos frear e acelerar mas ele não precisa ter os métodos de características como por exemplo carro elétrico ou carro a gasolina.

  Exemplo:

No exemplo acima, está mostrando a classe Livro que tem os métodos devolver e emprestar mas por exemplo não tem o método CapaDura pois é

uma verificação que não precisamos fazer por ser algo secundário, que não é para a funcionalidade do nosso código.

CONCLUSÃO

Durante o desenvolvimento dessa avaliação, aprendi como integrar de forma mais simples e prática as classes umas às outras, métodos de pesquisas em arrays de objetos como o FirstOrDefault que achei muito útil para as verificações de lista.

Também consegui fixar melhor o intuito das entidades e classes abstratas, fiquei muito mais confiante em minhas habilidades e conhecimentos na utlização de C# para desenvolvimento.

BIBLIOGRAFIA

POO: o que é programação orientada a objetos? Disponível em: <https://www.alura.com.br/artigos/poo-programacao-orientada-a-objetos?srsltid=AfmBOopuW8HeY>. Acesso em: 23 out. 2024.

COODESH, E. O que é POO em programação? Disponível em: <https://coodesh.com/blog/dicionario/o-que-e-poo-em-programacao/>. Acesso em: 23 out. 2024