Nome : Maria Luísa Pires Soares
Curso: Análise e desenvolvimento de sistemas
Cadeira: Laboratório de programação
Data: 24/10/2024

Introdução

A programação orientada a objetos (POO) é um paradigma amplamente utilizado no desenvolvimento de software. Ela se baseia na modelagem de sistemas complexos através de objetos, que combinam dados e comportamentos. Isso facilita a organização do código, a reutilização e a manutenção, tornando-se uma prática essencial na engenharia de software moderna.

Neste trabalho, será apresentado um sistema de gerenciamento de uma biblioteca digital desenvolvido com base nos quatro pilares da POO: encapsulamento, herança, polimorfismo e abstração. O sistema permite que usuários registrem empréstimos de livros e gerenciem seu histórico, utilizando conceitos fundamentais da POO para garantir um código modular e eficiente.

Encapsulamento

O encapsulamento é um dos pilares fundamentais da POO, e refere-se à capacidade de esconder os detalhes de implementação de um objeto e expor apenas as funcionalidades necessárias. Isso promove a proteção dos dados e impede que eles sejam acessados diretamente, permitindo que sejam manipulados por métodos controlados.

No nosso projeto de biblioteca digital, o encapsulamento foi aplicado na classe `Livro`. Os atributos `titulo`, `autor` e `statusEmprestimo` são privados, acessíveis apenas por meio de métodos específicos da classe.

No exemplo acima, o status de empréstimo (`statusEmprestimo`) é um detalhe de implementação que não pode ser modificado diretamente de fora da classe, garantindo que um livro só possa ser emprestado ou devolvido por meio dos métodos apropriados.

Herança

A herança é outro pilar essencial da POO, permitindo que classes derivem de outras classes, herdando seus atributos e métodos. Isso facilita a reutilização de código e a criação de hierarquias lógicas entre objetos.

No nosso projeto, a classe `Usuario` é a classe base, e dela derivamos a classe `UsuarioPremium`. O `UsuarioPremium` tem os mesmos atributos e comportamentos de `Usuario`, mas com permissões adicionais, como a possibilidade de pegar mais livros emprestados simultaneamente.

Nesse caso, `UsuarioPremium` herda o comportamento de `Usuario`, mas redefine o método `emprestarLivro` para permitir que usuários premium peguem até 10 livros emprestados, em vez de 5.

Polimorfismo

O polimorfismo permite que diferentes classes que herdam de uma mesma classe base possam ser tratadas de forma uniforme, mas com comportamentos diferentes. Isso é possível por meio da sobrescrita de métodos, como visto no exemplo anterior.

No nosso projeto, o polimorfismo é usado quando o sistema trata `Usuario` e `UsuarioPremium` da mesma forma, mas eles têm comportamentos diferentes ao pegar livros emprestados.

Exemplo de código:

O método `registrarEmprestimo` pode receber tanto um `Usuario` quanto um `UsuarioPremium`, mas o comportamento será diferente, já que o `UsuarioPremium` pode pegar mais livros emprestados.

Abstração

A abstração é o processo de simplificação de um sistema complexo, focando nos aspectos essenciais e ignorando detalhes irrelevantes para o contexto. Na POO, isso é feito por meio de classes abstratas ou interfaces, que definem contratos de comportamento para as classes concretas que as implementam.

No projeto, a abstração foi aplicada ao criar a interface `Emprestavel`, que define o comportamento comum a todos os itens que podem ser emprestados na biblioteca, como livros e revistas.

Exemplo de código:

A interface `Emprestavel` garante que qualquer item que possa ser emprestado (como `Livro` ou `Revista`) implemente os métodos `emprestar` e `devolver`, permitindo que o sistema os trate de forma uniforme.

Conclusão

O desenvolvimento do projeto de gerenciamento de biblioteca digital proporcionou uma compreensão profunda dos pilares da programação orientada a objetos. A aplicação de conceitos como encapsulamento, herança, polimorfismo e abstração resultou em um código modular, reutilizável e de fácil manutenção. Esses princípios são fundamentais para o desenvolvimento de sistemas robustos e escaláveis, e sua aplicação prática neste projeto demonstrou como a POO facilita a organização e clareza no código.

Referências Bibliográficas

- DEITEL, H.; DEITEL, P. **Java: Como Programar**. 10. ed. São Paulo: Pearson, 2015.
- SIERRA, K.; BATES, B. **Use a Cabeça! Java**. 2. ed. Rio de Janeiro: Alta Books, 2015.
- HORSTMANN, C. **Core Java Volume I: Fundamentals**. 11. ed. New York: Prentice Hall, 2018.