Capitulo 6 resumo:

Padrões de projeto são descrições das comunicações entre objetos e classes para poder resolver o problema em um contexto particular, saber esse padrões ajuda a implementa o codigo e também a testa-lo, melhorando a flexibilidade dele mesmo.

Exitem 23 padrões de projeto que podemos separar em 3 categorias principais, creational pattern (flexibilidade nas soluções e criações do objeto), Structural pattern (flexibiliza soluções para compor classes e objetos) Behavioral pattern (flexibilizar solução para interações e divisões de responsabilidade da classe).

Como dito existem diversos padrões que entram nas 3 categoria, na categoria creational temos os padrões

Creational pattern

- Padrão Fábrica -> A implementação do padrão Factory utiliza o princípio Prefira Interfaces a Classes, também conhecido como Princípio da Inversão de Dependência, onde todas as funções usam a interface Channel para manipular os objetos criados pelo padrão Factory Method, e um método de fábrica estático encapsula e confina as novas chamadas, facilitando a modificação do tipo de canal ao alterar um único elemento de código

- Padrão Singleton -> garante que uma classe tenha no máximo uma instância, e sua aplicação em uma classe Logger para evitar a proliferação de objetos Logger e ter uma única instância usada em todo o sistema para registrar eventos.

Padrões estruturais

- Padrão Proxy -> envolve o uso de um objeto intermediário, conhecido como proxy, entre o objeto base, permitindo a adição de funcionalidades, como cache, sem modificar o objeto base.

- Padrão Adaptador ->O padrão de projeto Adapter, também conhecido como Padrão Adapter, é usado para converter a interface de uma classe em outra, permitindo que uma interface unificada seja usada com diferentes classes

- Padrão Fachada -> O padrão de projeto Facade é usado para fornecer uma interface simplificada para um sistema complexo.alcança isso ao oferecer uma única classe que encapsula a complexidade do sistema, permitindo que os usuários interajam apenas com a classe Facade, enquanto a complexidade interna permanece oculta por trás dela, simplificando a interface e reduzindo a complexidade do código.

- Padrão Decorador -> Decorator, que oferece uma alternativa à herança ao empregar composição para adicionar dinamicamente essas funcionalidades a objetos base.

Padrões de comportamento

- Padrão Strategy -> O padrão Strategy acomoda múltiplos algoritmos de ordenação, permitindo que o algoritmo de ordenação seja parametrizado e intercambiável, o que é alcançado ao definir uma classe abstrata SortStrategy e classes concretas como QuickSortStrategy e ShellSortStrategy.

- Padrão Observador -> O padrão de projeto Observer é uma solução recomendada para implementar um relacionamento um-para-muitos entre objetos sujeitos e observadores, onde o estado de um sujeito muda e todos os seus observadores são notificados

- Padrão Template Method -> O padrão de projeto Template Method é usado para definir um modelo para um algoritmo em uma classe abstrata, que pode ser herdado por suas subclasses, permitindo que elas adaptem o modelo às suas necessidades específicas sem alterar sua estrutura fundamental

- Padrão Visitor -> uma interface Visitor é implementada por classes para realizar operações específicas em diferentes tipos de objetos.