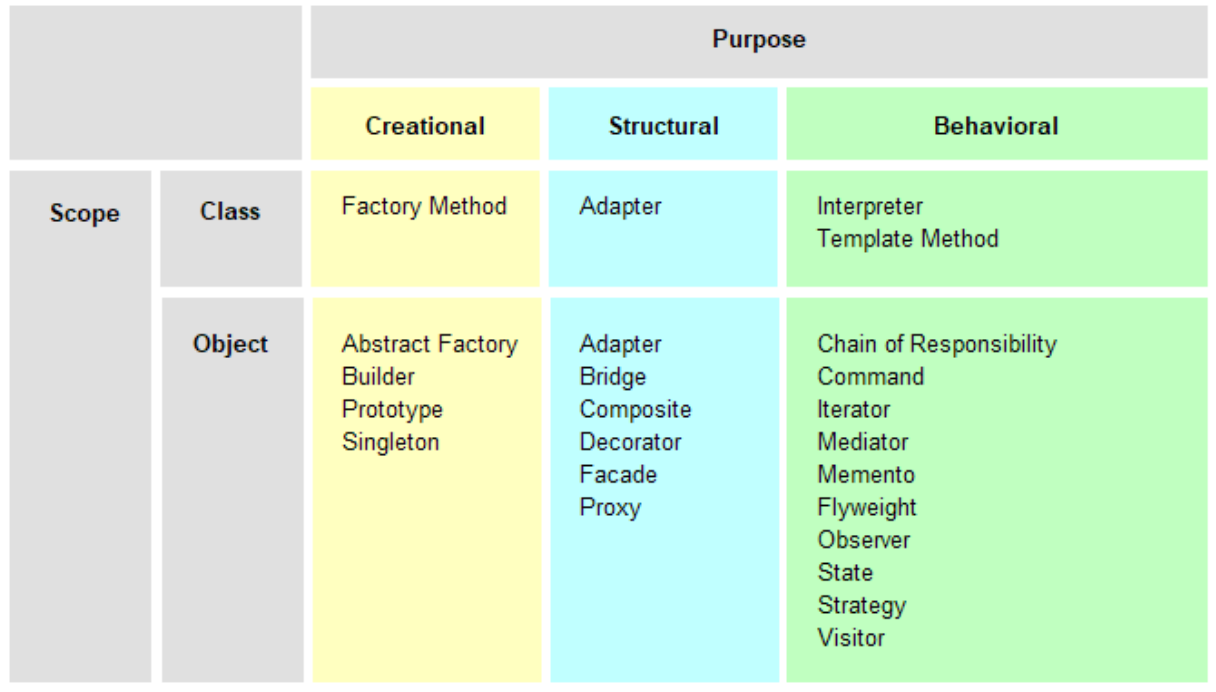
Design Patterns:

- **Creational Patterns:** Used to construct objects such that they can be decoupled from their implementing system.  
- **Structural Patterns:** Used to form large object structures between many disparate objects.  
- **Behavioral** **Patterns:** Used to manage algorithms, relationships, and responsibilities between objects.



Padrões Criacionais (Creational Patterns)

Os padrões criacionais fornecem vários mecanismos de criação de objetos, que aumentam a flexibilidade e a reutilização do código existente.

Factory mode

Factory mode (br: modelo de fábrica) é um padrão de design criacional que fornece uma interface para criar objetos em uma superclasse, mas permite que as subclasses alterem o tipo de objeto que será criado.

+Detalhes

Abstract Factory

Abstract Factory (br: fábrica abstrata) é um padrão de design criacional que permite produzir famílias de objetos relacionados sem especificar suas classes concretas.

+Detalhes

Builder

Builder (br: construtor) é um padrão de design criacional que permite construir objetos complexos passo a passo. O padrão permite produzir diferentes tipos e representações de um objeto usando o mesmo código de construção.

+Detalhes

Singleton

Singleton é um padrão de design criacional que permite garantir que uma classe tenha apenas uma instância, enquanto fornece um ponto de acesso global a essa instância.

+Detalhes

Prototype

Prototype é um padrão de design criacional que permite copiar objetos existentes sem tornar seu código dependente de suas classes.

+Detalhes

Padrões estruturais (Structural Design Patterns)

Os padrões estruturais explicam como montar objetos e classes em estruturas maiores, mantendo essas estruturas flexíveis e eficientes.

Adapter

Adapter (br: adaptador) é um padrão de design estrutural que permite a colaboração de objetos com interfaces incompatíveis.

+Detalhes

Bridge

Bridge (br: ponte) é um padrão de design estrutural que permite dividir uma classe grande ou um conjunto de classes estreitamente relacionadas em duas hierarquias separadas — abstração e implementação — que podem ser desenvolvidas independentemente uma da outra.

Composite

Composite (br: Composto) é um padrão de design estrutural que permite compor objetos em estruturas de árvores e trabalhar com essas estruturas como se fossem objetos individuais.

Decorator

Decorator (br: decorador) é um padrão de design estrutural que permite anexar novos comportamentos aos objetos, colocando-os dentro de objetos especiais do invólucro que contêm os comportamentos.

Facade

Facade (br: Fachada) é um padrão de design estrutural que fornece uma interface simplificada para uma biblioteca, uma estrutura ou qualquer outro conjunto complexo de classes.

Flyweight

Flyweight é um padrão de design estrutural que permite ajustar mais objetos à quantidade disponível de RAM, compartilhando partes comuns do estado entre vários objetos, em vez de manter todos os dados em cada objeto.

Proxy

Proxy é um padrão de design estrutural que permite fornecer um substituto ou espaço reservado para outro objeto. Um proxy controla o acesso ao objeto original, permitindo que você execute algo antes ou depois que a solicitação chega ao objeto original.

Padrões Comportamentais (Behavioral Design Patterns)

Os padrões de design comportamental preocupam-se com algoritmos e a atribuição de responsabilidades entre objetos.

Chain of Responsibility

Chain of Responsibility (br: Cadeia de responsabilidade) é um padrão de design comportamental que permite passar solicitações ao longo de uma cadeia de manipuladores. Ao receber uma solicitação, cada manipulador decide processar a solicitação ou passá-la para o próximo manipulador na cadeia.

Command

Command (br: comando) é um padrão de design comportamental que transforma uma solicitação em um objeto independente que contém todas as informações sobre a solicitação. Essa transformação permite parametrizar métodos com diferentes solicitações, atrasar ou enfileirar a execução de uma solicitação e oferecer suporte a operações que podem ser desfeitas.

Iterator

Iterador (br: iterador) é um padrão de design comportamental que permite percorrer elementos de uma coleção sem expor sua representação subjacente (lista, pilha, árvore etc.).

Mediator

Mediator (br: Mediador) é um padrão de design comportamental que permite reduzir dependências caóticas entre objetos. O padrão restringe as comunicações diretas entre os objetos e os força a colaborar apenas por meio de um objeto mediador.

Memento

Memento (br: lembrança) é um padrão de design comportamental que permite salvar e restaurar o estado anterior de um objeto sem revelar os detalhes de sua implementação.

Observer

Observer (br: Observador) é um padrão de design comportamental que permite definir um mecanismo de assinatura para notificar vários objetos sobre quaisquer eventos que ocorram no objeto que estão observando.

State

State (br: estado) é um padrão de design comportamental que permite que um objeto altere seu comportamento quando seu estado interno for alterado. Parece que o objeto mudou de classe.

Strategy

Strategy (br: Estratégia) é um padrão de design comportamental que permite definir uma família de algoritmos, colocar cada um deles em uma classe separada e tornar seus objetos intercambiáveis.

Template Method

Template Method (br: Método do Modelo) é um padrão de design comportamental que define o esqueleto de um algoritmo na superclasse, mas permite que as subclasses substituam etapas específicas do algoritmo sem alterar sua estrutura.

Visitor

Visitor (br: Visitante) é um padrão de design comportamental que permite separar algoritmos dos objetos nos quais eles operam.

Nós próximos posts irei falar mais sobre design patterns e pretendo fazer um exemplo de cada um deles.