

Design Patterns- Introdução

2022

Prof. Sergio Bonato

Conceitos Básicos (1/2)

- Padrões de Projeto (Design Patterns) são soluções usadas e experimentadas para problemas comuns do desenvolvimento de software. Os Design Patterns foram idealizados por Kent Beck, o mesmo da Extreme Programming, na década de 80.



Conceitos Básicos (2/2)

- O primeiro pattern documentado do qual se tem notícia é o MVC, publicado em 88 por Karner & Pope no Journal of OOP como um padrão para a organização do código em SmallTalk. Mas o assunto ganhou fama mesmo quando Gamma, Helm, Johnson e Vlissides, a Gang of Four (GoF), publicou um livro com 23 design patterns, inspirando-se no trabalho do arquiteto (de prédios, não de software) Christopher Alexander.
- Os patterns devem ser descritos de uma maneira clara e organizada para facilitar seu uso. A GoF sugere o seguinte modelo, que não é obrigatório. Há outros modelos de descrição de patterns, mas todos contêm mais ou menos as mesmas informações.

Modelo para Descrição de Patterns (1/3)

- **Nome do Pattern:** o nome deve indicar a essência do pattern.
- **Classificação:** quanto ao propósito, o GoF classifica os patterns como criacionais (lidam com criação de objetos), estruturais (tem a ver como os objetos se relacionam) e comportamentais (como os objetos interagem e como é feita a distribuição de responsabilidades); quando ao escopo, os patterns se classificam em de classe e de objetos.
- **Intenção:** um texto curto que diz o que o pattern faz, explica sua razão de ser e sua intenção e, principalmente, qual o problema que ele pretende resolver.
- **Apelido:** outros nomes pelos quais o pattern é conhecido.

Modelo para Descrição de Patterns (2/3)

- **Motivação:** um cenário que ilustra um problema de design e como as classes e objetos se estruturam no pattern para resolver o problema.
- **Aplicabilidade:** as situações em que o pattern pode ser aplicado.
- **Estrutura:** uma representação gráfica das classes e objetos no pattern, usando UML; sempre há o diagrama de classes e, geralmente, o de sequencia.
- **Participantes:** as classes e objetos que participam do design pattern e suas responsabilidades.
- **Colaborações:** como os participantes colaboram na execução de suas responsabilidades.

Modelo para Descrição de Patterns (3/3)

- **Consequências:** quais são as escolhas que devem ser feitas para o uso deste pattern.
- **Implementação:** armadilhas, dicas e técnicas para o implementador.
- **Código exemplo:** fragmentos de código mostrando o uso do pattern em C++ ou Java. Usos conhecidos: exemplos dos patterns encontrados em sistemas reais.
- **Patterns relacionados:** quais patterns estão relacionados com este, mostrando semelhanças e diferenças.

Por que usar Patterns? [JOSHI 16]

- **Flexibilidade**: todo pattern visa reduzir o acoplamento; isso torna o código flexível.
- **Reuso**: código com alta coesão e baixo acoplamento tende a ser reutilizável.
- **Vocabulário compartilhado**: ao dar um nome para o pattern fica mais fácil todo mundo saber do que se está falando.
- **Captura de boas práticas**: você pode criar seus próprios patterns e compartilhá-los usando o modelo de descrição.

Tipos de Patterns do GoF [GAMMA 00]

- **Criacionais**: usados no processo de instanciação de objetos.
- **Estruturais**: definem como as classes são compostas para formar estruturas maiores.
- **Comportamentais**: tratam de algoritmos e da atribuição de responsabilidades entre os objetos.

Tabela de Patterns do GoF

os padrões sublinhados serão detalhados nesta disciplina conforme a referência [JOSHI 16] e [Freeman et al 04]

Frequência de Uso¹

- Alta
- Média Alta
- Média
- Média Baixa
- Baixa

	Criacionais		Estruturais		Comportamentais
●●●●●	<u>Abstract Factory</u>	●●●●●	<u>Adapter</u>	●●●●●	Chain of Responsibility
●●●●●	Builder	●●●●●	Bridge	●●●●●	<u>Command</u>
●●●●●	<u>Factory Method</u>	●●●●●	<u>Composite</u>	●●●●●	Interpreter
●●●●●	Prototype	●●●●●	<u>Decorator</u>	●●●●●	<u>Iterator</u>
●●●●●	<u>Singleton</u>	●●●●●	<u>Facade</u>	●●●●●	Mediator
		●●●●●	Flyweight	●●●●●	Memento
		●●●●●	<u>Proxy</u>	●●●●●	<u>Observer</u>
				●●●●●	State
				●●●●●	<u>Strategy</u>
				●●●●●	<u>Template Method</u>
				●●●●●	Visitor
1. De acordo com o site https://www.dofactory.com/net/design-patterns (2019)					

Bibliografia

[GAMMA 00] GAMMA et al.; Padrões de Projeto: soluções reutilizáveis de software orientado a objetos. 1ª Edição. Bookman. 2000.