Padrão de Projeto: Proxy

Trabalho de Engenharia de Software II

Análise e Aplicação de Padrões GoF

Introdução aos Padrões GoF

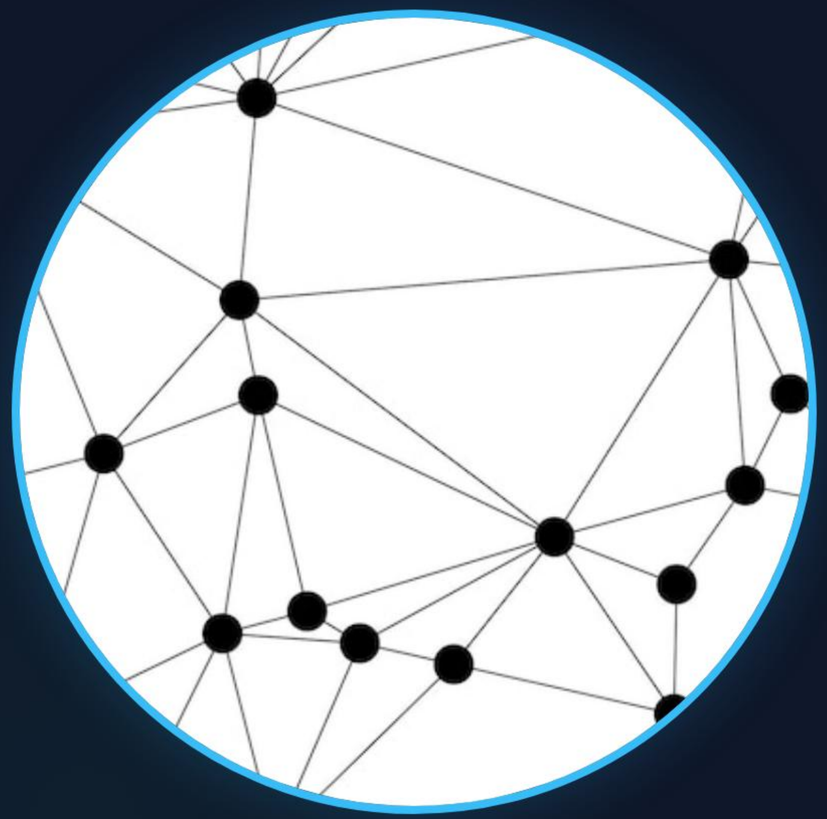
Soluções Reutilizáveis

Os padrões de projeto da Gang of Four (GoF) são soluções consolidadas para problemas recorrentes no desenvolvimento de software orientado a objetos.

Benefícios Chave

A aplicação destes padrões melhora significativamente a comunicação entre desenvolvedores, promove a flexibilidade do código e facilita a manutenção a longo prazo.

Categoria: Estrutural



Organização e Composição

O padrão Proxy pertence à família dos **Padrões Estruturais**.

Estes padrões se preocupam com a forma como classes e objetos são compostos para formar estruturas maiores, mantendo a flexibilidade e a eficiência do sistema.

Propósito do Padrão

> *"Fornecer um substituto ou espaço reservado para outro objeto para controlar o acesso a ele."*

Adiciona comportamentos como controle de acesso, lazy loading, logging ou cache.

O Problema a Resolver

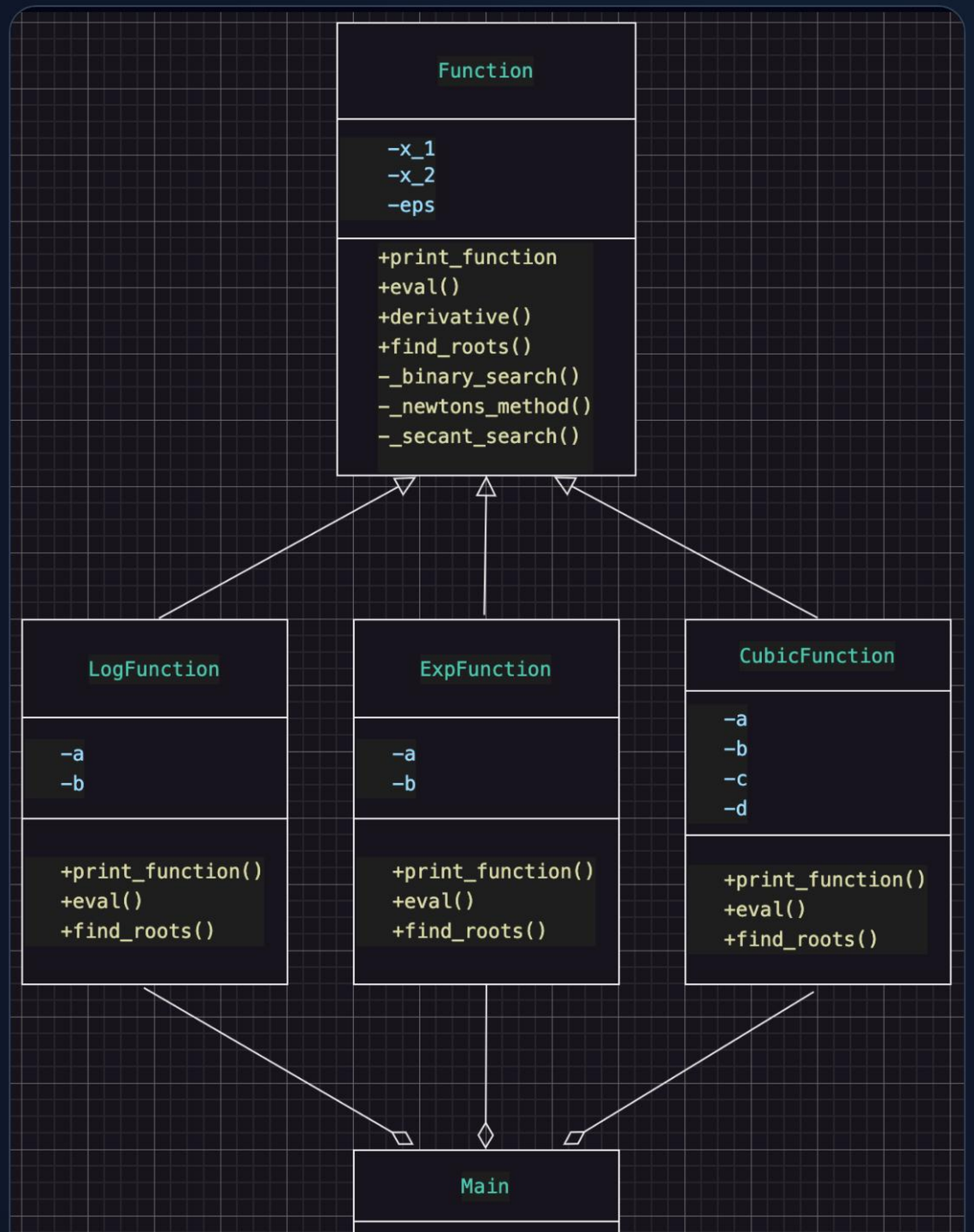
Em sistemas complexos, frequentemente precisamos acessar objetos que são pesados (consomem muita memória), sensíveis (requerem segurança) ou estão distantes (em outro servidor).

O acesso direto a estes objetos pelo código cliente pode causar problemas de performance, vulnerabilidades de segurança ou acoplamento desnecessário.

Intermediação Transparente

O Proxy implementa a mesma interface que o objeto real (RealSubject).

- O cliente interage com o Proxy como se fosse o objeto real.
- O Proxy intercepta a chamada.
- O Proxy executa lógicas adicionais (ex: verifica permissão).
- O Proxy repassa a chamada ao objeto real, se apropriado.



Tipos Comuns de Proxy

Proxy Virtual

Cria objetos pesados apenas quando necessário (Lazy Load).

Proxy de Proteção

Controla o acesso com base em permissões de usuário.

Proxy Remoto

Representa localmente um objeto que está em outro servidor.

Proxy de Cache

Armazena resultados de operações caras para evitar reprocessamento.

Principais Vantagens

- **Controle Total:** Gerencia o ciclo de vida e o acesso ao objeto real de forma transparente.
- **Performance:** Otimiza recursos através do carregamento sob demanda (lazy loading).
- **Segurança:** Protege o objeto real de clientes não autorizados.
- **Princípio Open/Closed:** Permite adicionar novos proxies sem alterar o serviço ou o cliente.

Desvantagens e Cuidados

Complexidade: Introduz novas classes e interfaces, aumentando a complexidade do código.

Latência: A camada intermediária pode adicionar atrasos na resposta (especialmente em Proxies Remotos).

Manutenção: Alterações na interface do objeto real exigem atualizações no Proxy.

Confusão: Pode ser confundido conceitualmente com o padrão Decorator, embora as intenções sejam diferentes.

Conclusão

O padrão Proxy é essencial para sistemas que demandam **controle, segurança e eficiência**. Ele oferece uma maneira elegante de gerenciar o acesso a objetos complexos, mantendo o princípio da responsabilidade única.

Referências Bibliográficas

Literatura Base

Gamma, Erich et al.
Design Patterns: Elements of Reusable Object-Oriented Software (GoF). Addison-Wesley, 1994.

Recursos Online

Refactoring Guru
https://refactoring.guru/pt-br/design-patterns/proxy

Material da Disciplina
Slides de Engenharia de Software II

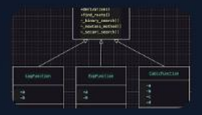
Image Sources

https://img.freepik.com/premium-vector/abstract-polygonal-mesh-with-black-nodes-thin-lines-creating-minimalist-geometric-network-pattern_1282075-5152.jpg

Source: www.freepik.com

https://img.freepik.com/premium-photo/technology-background-abstract-cyber-security-system-pad-lock-with-finger-print-icon_824086-1640.jpg

Source: www.freepik.com



https://i.sstatic.net/UotAt.png

Source: stackoverflow.com

https://img.freepik.com/premium-vector/glowing-shield-cybersecurity-icon-futuristic-digital-protection-data-security-network-safety-concept-with-bright-neon-shield-dark-background_145776-164503.jpg

Source: www.freepik.com