Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas
Prof. Felippe Scheidt – IFPR – Campus Foz do Iguaçu

Frameworks

Padrões de projetos

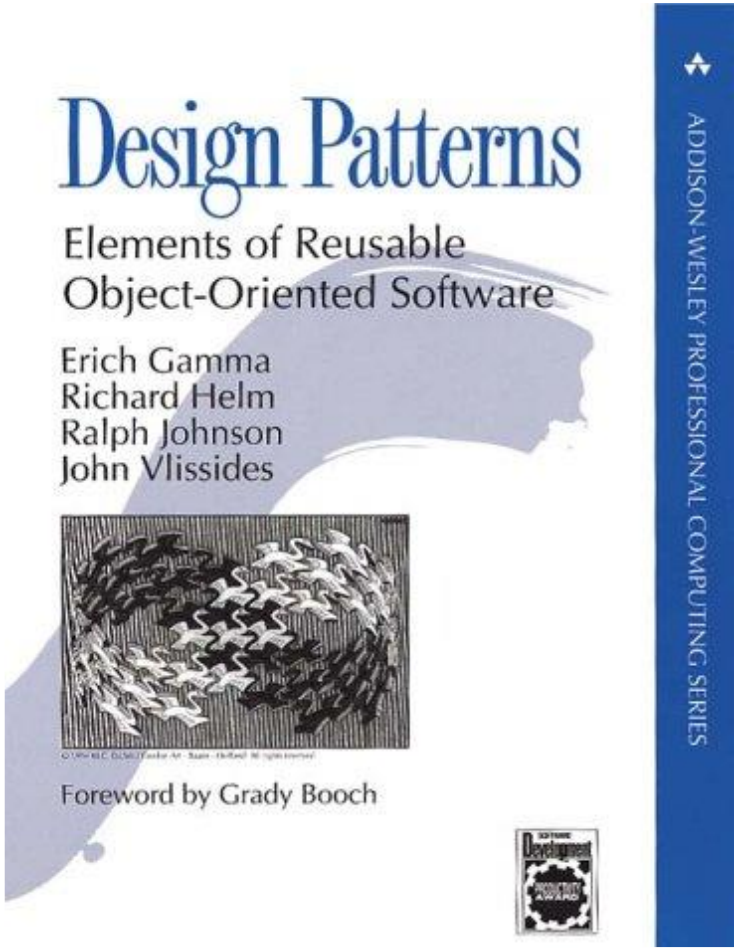
Inspiração

- A ideia de padrões foi apresentada por Christopher Alexander em 1977 no contexto de Arquitetura (de prédios e cidades):

Cada padrão descreve um problema que ocorre repetidamente de novo e de novo em nosso ambiente, e então descreve a parte central da solução para aquele problema de uma forma que seja possível usar esta solução um milhão de vezes, sem nunca implementa-la duas vezes da mesma forma.

Gang of Four (GoF)

- E. Gamma and R. Helm and R. Johnson and J. Vlissides. **Design Patterns - Elements of Reusable Object-Oriented Software**. Addison-Wesley, 1995.



Gang of Four (GoF)

- Introduziu um vocabulário comum para conversar sobre projetos de software.
- Soluções que não tinham nome passaram a ter nome.
- Ao invés de discutir um sistema em termos de pilhas, filas, árvores e listas ligadas, passou-se a falar de coisas de muito mais alto nível como Fábricas, Fachadas, Observador, Estratégia, etc.
- Mas o livro foi lançado em C++ para que o impacto junto à comunidade de Ciência da Computação fosse maior.

Objetivos

- Reaproveitar uma solução
- Lembrar de uma solução anterior (*déjà vu*)
- Identificar, explicar e analisar uma solução.
- Padrões de projetos = registro de uma experiência de projeto (desenho) de um software orientado a objetos.
- A partir de uma descrição detalhada, outras pessoas podem fazer uso dessa solução.
- Torna mais fácil a tarefa de reutilizar soluções de projeto.
- Apenas são ditos padrões de projetos aqueles que aparecem em diversos projetos.

Catálogo de soluções

- Um padrão agrupa o conhecimento de uma pessoa muito experiente em um determinado assunto de uma forma que este conhecimento possa ser transmitido para outras pessoas menos experientes.
- Desde 1995, o desenvolvimento de software passou a ter o seu primeiro catálogo de soluções para projeto de software: o livro GoF.

O Formato de um padrão

- Todo padrão inclui:
  - Nome
  - Problema (explica o problema e seu contexto)
  - Solução
  - Consequências (resultados esperados)

O Formato dos padrões no GoF

- **Nome**
  - um bom nome é essencial para que o padrão caia na boca do povo
- **Classificação**
  - Criação, estrutural, comportamental
- **Objetivo / Intenção**
- **Motivação**
  - um cenário mostrando o problema e a necessidade da solução
- **Aplicabilidade**
  - como reconhecer as situações nas quais o padrão é aplicável
- **Estrutura**
  - uma representação gráfica da estrutura de classes do padrão (diagrama)
- **Participantes**
  - as classes e objetos que participam e quais são suas responsabilidades

O Formato dos padrões no GoF

- **Colaborações**
  - como os participantes colaboram para exercer as suas responsabilidades
- **Consequências**
  - vantagens e desvantagens, *trade-offs* (conflitos de decisão)
- **Implementação**
  - com quais detalhes devemos nos preocupar quando implementamos o padrão, levando em consideração aspectos específicos de cada linguagem
- **Exemplo de Código**
- **Usos Conhecidos**
  - exemplos de sistemas reais de domínios diferentes onde o padrão é utilizado
- **Padrões Relacionados**
  - quais outros padrões devem ser usados em conjunto com esse
  - quais padrões são similares a este, quais são as diferenças

Tipos de Padrões de Projeto

- Categorias de Padrões do GoF
  - Padrões de **Criação**: relacionados à criação de objetos;
  - Padrões **Estruturais**: tratam das associações entre classes e objetos.
  - Padrões **Comportamentais**: tratam das interações e divisões de responsabilidades entre as classes ou objetos.

Padrões de criação

1. Abstract Factory
2. Builder
3. Factory Method
4. Prototype
5. Singleton

Padrões estruturais

1. Adapter
2. Bridge
3. Composite
4. Decorator
5. Facade
6. Flyweight
7. Proxy

- *associações entre classes e objetos;

Padrões comportamentais

1. Chain of Responsibility
2. Command
3. Interpreter
4. Iterator
5. Mediator
6. Memento
7. Observer
8. State
9. Strategy
10. Template Method
11. Visitor

- *interações e divisões de responsabilidades entre as classes ou objetos.

Estes padrões são OO...

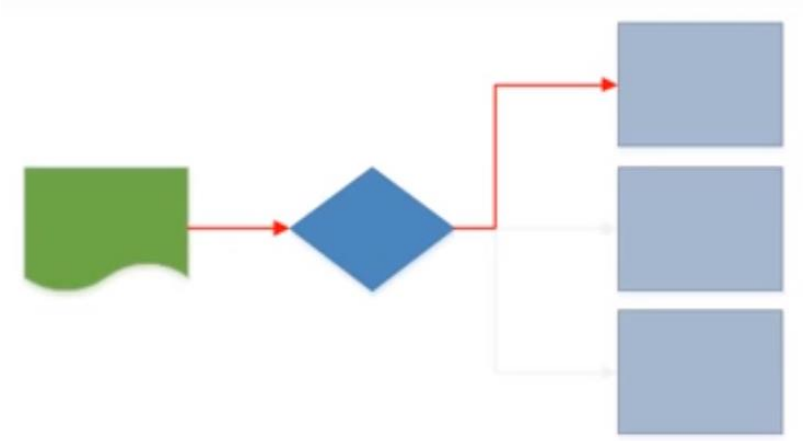
- ...então é preciso ter bem claro determinados conceitos:
- **Abstração;**
- **Encapsulamento**;
- **Polimorfismo**;
- **Herança**;

Padrão Strategy

- O padrão Strategy é um padrão comportamental, que permite que seja decidido o curso de ação que o programa deve tomar, baseado no contexto de execução da aplicação. Para isso, cada tipo de algoritmo é encapsulado em diferentes classes, e em tempo de execução será decidida qual estratégia será utilizada.
- O **Strategy** permite que o algoritmo mude (varie) independentemente dos clientes que o usam.

Strategy

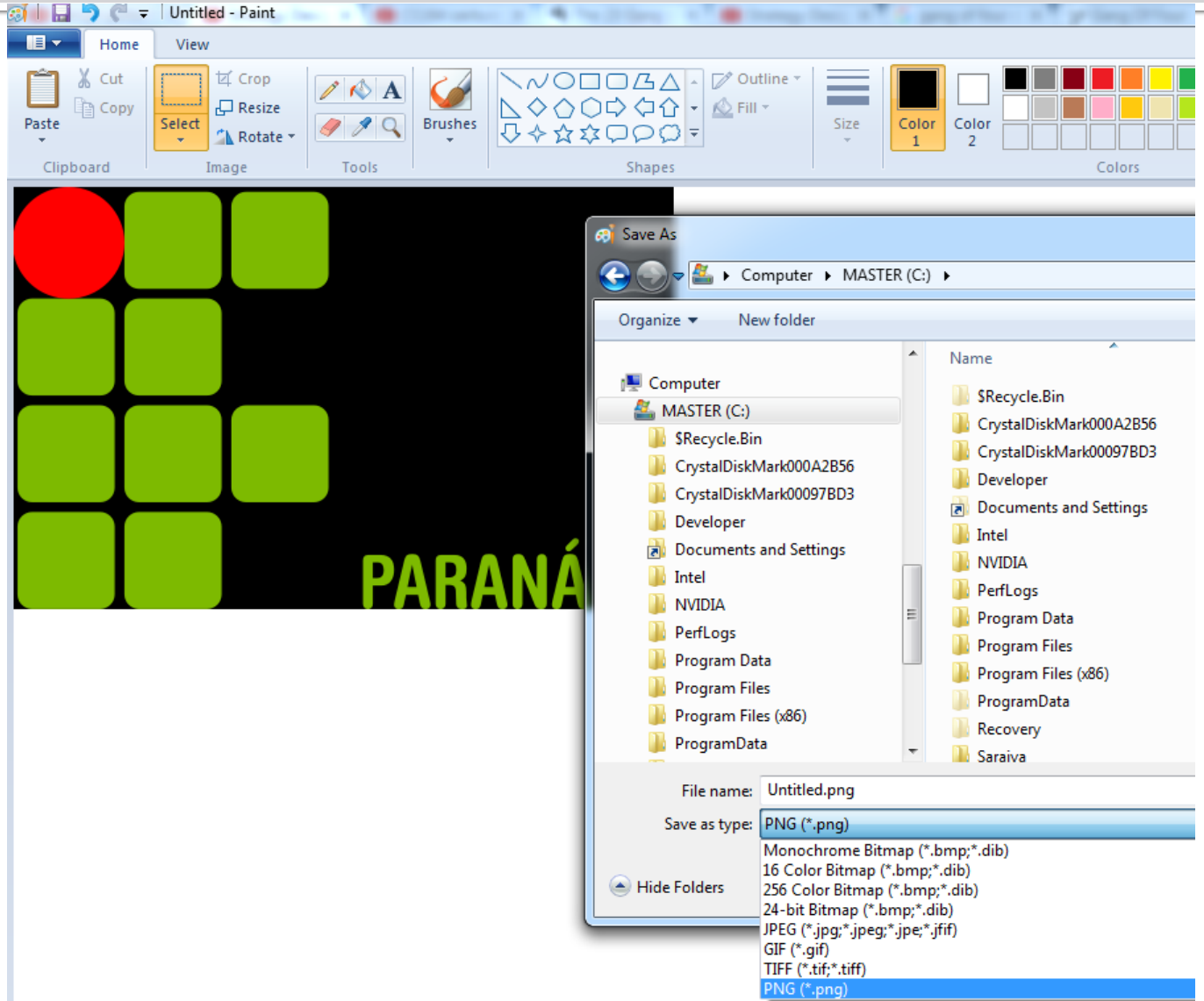
- Decisão sobre qual estratégia utilizar
- Alterar a forma que um determinado trecho de código funciona.
- Existem vários algoritmos para solucionar o problema. Agrupa-se cada estratégia sobre uma mesma interface.
- Assim, em tempo de execução podemos decidir qual estratégia utilizar.



Exemplo de cenário

- Salvar uma imagem
- De acordo com o tipo da imagem, escolher a estratégia adequada.
  - png
  - gif
  - bmp
  - jpg

Exemplo de cenário



Atividade

- Pesquise sobre o padrão strategy e proponha uma implementação usando esse padrão.