Design Patterns
STRATEGY

EMERSON BARROS DE MENESES

Breve Histórico Sobre Design Patterns

A origem dos Design Patterns (Padrões de Desenho ou ainda Padrões de Projeto) vem do trabalho de um arquiteto chamado Christopher Alexander, no final da década de 70. Ele escreveu dois livros, inicialmente, A Pattern Language [Alex77] e A Timeless Way of Building [Alex79], nos quais ele exemplificava o uso e descrevia seu raciocínio para documentar os padrões. Em 1995, um grupo de quatro profissionais escreveu e lançou o livro "Design Patterns: Elements of Reusable Object-Oriented Software" [Gamma95] e traduzido para “Padrões de Projeto: soluções reutilizáveis de software orientado a objetos” [2000], um catálogo com 23 padrões de desenho (design patterns). Os autores: Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, ficaram mais conhecidos como A Gangue dos Quatro (Gang Of Four ou GoF), considerados os maiores entusiastas dos Design Patterns. Cujo mesmo livro foi utilizado na criação deste trabalho.

Breve resumo sobre “o que é um Design Patterns”?

Os Design Patterns são padrões de classes e de relacionamentos entre as mesmas que aparecem de forma freqüente em projetos de software. Tais padrões são categorizados para atender a soluções específicas. Sua utilização é uma atividade que simplifica a reutilização de software. Os padrões aparecem em situações especiais dentro de um sistema como “descrições de objetos e classes comunicantes que são customizadas um contexto particular”. Este descreve uma solução comprovada para um problema, são mais abstratos, menores e menos específicos que frameworks que por sua vez é um conjunto de classes que pode ser utilizado para um tipo específico de projeto de software (análise de domínio, biblioteca de reuso).

Uma primeira visualização (entendimento) corresponde à tríade modelo, vista e controlador (MVC). Procuram estabelecer uma distinção rígida entre estes elementos de modo a torná-los independentes. Ou seja, modelo, vista e controlador podem ser modificados de forma independente (desacoplamento).

Patterns são dispositivos que permitem que os programas compartilhem conhecimento sobre o seu desenho. Quando programamos, encontramos muitos problemas que ocorrem, ocorreram e irão ocorrer novamente. A questão que nos perguntamos agora é como nós vamos solucionar este problema desta vez? Documentar um padrão (pattern) é uma maneira de poder reusar e possivelmente compartilhar informação que aprendeu sobre a melhor maneira de se resolver um problema de desenho de software. O catálogo de padrões do GoF (Gang Of Four), como dito, contém 23 padrões e está, basicamente, dividido em três seções: Creational (Criacional), Structural (Estrutural), Behavioral (Comportamental). Neste trabalho iremos elencar pontos referentes ao Padrão Criacional Abstract Factory.

Strategy (Design Patterns Creational)

Objetivo

- Definir uma família de algoritmos, encapsular cada um, e torná-los intercambiáveis

Escopo

- De Objeto

Propósito

- Comportamental

Também chamado de

- Policy

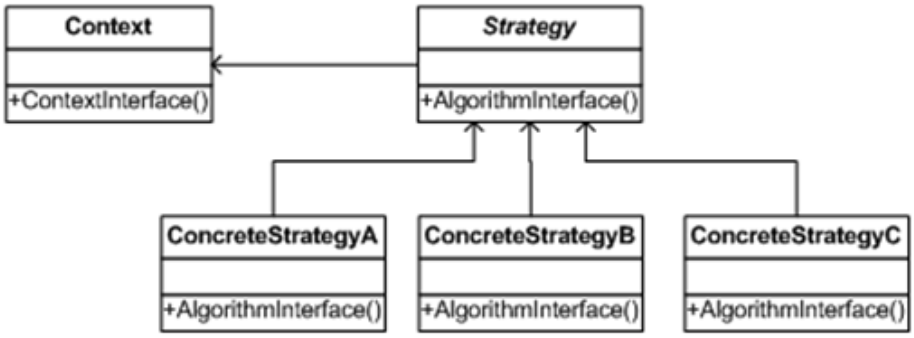
Resumo:

- Ele é útil nas situações nas quais é necessário trocar dinamicamente o algoritmo a ser usado em uma aplicação.
- Permite mudar os algoritmos independentemente dos clientes que os usam.

Quando Usar:

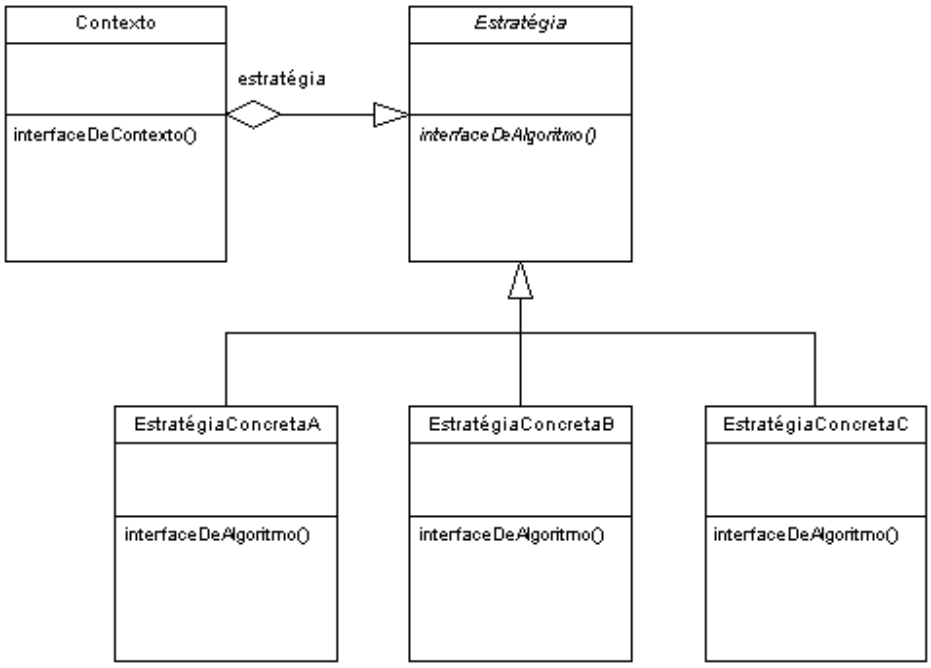
- Várias classes relacionadas têm apenas comportamento diferente
- Você precisa de variantes de um algoritmo
- Para esconder dos clientes os dados complexos que um algoritmo usa
- Quando se necessita de um algoritmo que trata de modos diferentes os dados submetidos a ele.

Estrutura genérica



Participantes: Strategy (controlador abstrato), ConcreteStrategy (controlador concreto), Context (modelo).

Mais detalhadamente seria:



Participantes:

Estratégia (Formatador)

- Classe abstrata para fatorar código comum entre os algoritmos

EstratégiaConcreta (FormatadorSimples)

- Implementa o algoritmo

Contexto (exemplo: ElementosAFormatar)

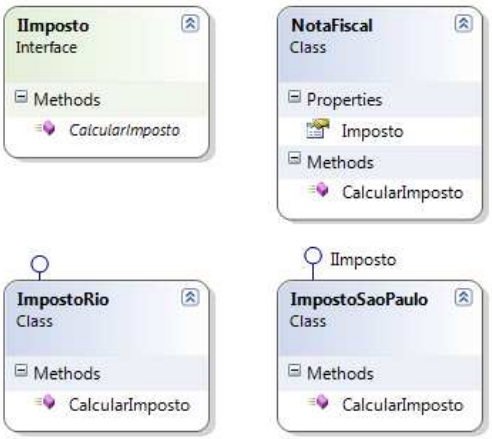
- É configurado com um objeto EstratégiaConcreta
- Pode definir uma interface para que a estratégia acesse seus dados

Como Funciona:

Encapsula o algoritmo controlador de uma estrutura ou de um conjunto de classes. Desta forma, as operações que manipulam os dados são armazenados em outras classes.

Além disso, cria um estrutura em separado, para abrigar as diferentes implementações do algoritmo

Exemplo de código: calculo das notas fiscais



Código da interface do imposto:

```
public interface IImposto

{

    decimal CalcularImposto(decimal valor);

}
```

Código da Classe Nota fiscal

```
public class NotaFiscal
{
public IImposto Imposto { get; set; }
    public decimal CalcularImposto(decimal valor)
    {
        return Imposto.CalcularImposto(valor);
    }
}
```

Classe concreta Imposto São Paulo

```
public class ImpostoSaoPaulo:IImposto
{
public decimal CalcularImposto(decimal valor)
    {
        return valor * 1.2M;
      }
}
```

Classe concreta Imposto Rio de Janeiro

```
public   class ImpostoRio: IImposto
```

```
{
public decimal CalcularImposto(decimal valor)
    {
        return valor * 1.2M;
    }
}
```

Código responsável pela chamada da nota fiscal

```
NotaFiscal notaFiscalDP = new NotaFiscal();
IImposto imposto = new ImpostoSaoPaulo();
notaFiscalDP.Imposto = imposto;
notaFiscalDP.CalcularImposto(55M);
```

Com o padrão strategy, quando houver algum tipo de mudança no formato do calculo fiscal, não precisamos fazer nenhuma alteração na classe NotaFiscal, é só criarmos uma nova classe, implementar a interface IImposto e na variável imposto, da classe NotaFiscal passar a nova classe, que a NotaFiscal fará o calculo da mesma.

Bibliografia:

http://pt.wikipedia.org/wiki/Strategy

http://www.dsc.ufcg.edu.br/~jacques/cursos/map/html/pat/strategy.htm

GAMMA, Erich; HELM, Richard; JOHNSON, Ralph; VLISSIDES, John. Padrões de Projeto: soluções reutilizáveis de software orientado a objetos. Ed. 1. Bookman, 2000. ISBN 9788573076103.

Eric Freeman; Use a cabeça! : padrões de projetos; tradução Andreza Gonçalves, Marcelo Soares e Pedro Cesar de Conti. Ed. 1. Alta Books, 2009.

Metsker, Steven John; Padrões de projeto em Java; Bookman, 2004, ISBN 85-363-0411-1

Design Patterns - STRATEGY

Por Emerson Barros de Meneses