Encontro de Tecnologia e Informática 2003
Instituto de Informática
UFG

Design Patterns e Reutilização de Software

Prof. Sérgio Teixeira de Carvalho
sergiocarvalho@anhanguera.edu.br

eti 2003

Design Patterns e Reutilização de Software

Conceituação

Descrição

Exemplos

Onde Encontrá-los ?

Experiência

Experiência

alguém já viveu situação semelhante !

transmitida com informalidade

difícil compartilhar

Em sistemas de software, podemos ‘reutilizar’ a experiência na solução de algum problema

Em sistemas de software, podemos 'reutilizar' a experiência na solução de algum problema

Como ?

design patterns

“ Um pattern descreve de forma padronizada um problema recorrente dentro de contextos específicos e apresenta um esquema genérico para sua solução “ [1]

design patterns

- documentam soluções a problemas recorrentes no desenvolvimento de sistemas de software
- permitem a reutilização da experiência no desenvolvimento de software
- descrevem uma solução para um problema em um contexto (formato *contexto-problema-solução*)

design patterns

- devem ser comprovados, ou seja, testados e experimentados previamente
- não são invenções
- capturam a evolução e aprimoramento das soluções
- podem ser utilizados com outros padrões, compondo *frameworks* ou arquiteturas

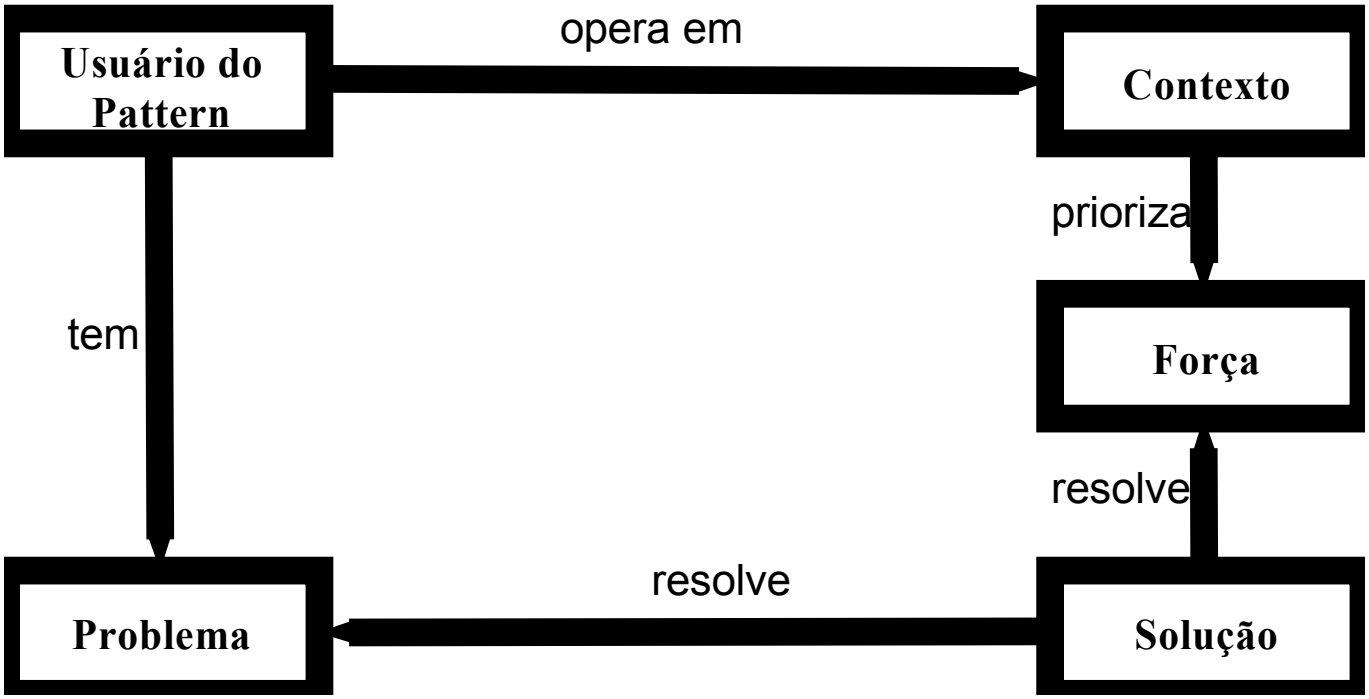
descrevendo um design pattern

- Nome
- Contexto
- Problema
- Forças
- Solução
- Conseqüências
- Patterns Relacionados
- Usos Conhecidos
- Código-Exemplo

descrevendo um design pattern

- Nome
- Contexto
- Problema
- Forças
- Solução
- Conseqüências
- Patterns Relacionados
- Usos Conhecidos
- Código-Exemplo

descrevendo um design pattern



a descrição de design patterns:

- deve fornecer uma descrição clara do problema e da solução proposta
- deve contemplar os detalhes necessários à sua implementação e considerar as conseqüências de sua aplicação
- deve descrever por que e como usá-los, em vez de documentar o que são

alguns design patterns [2]

- **Patterns de criação (*creational*)**
  - Abstract Factory, Builder, Factory Method, Prototype, Singleton
- **Patterns de Estrutura (*structural*)**
  - Adapter, Bridge, Composite, Decorator, Facade, Flyweight, Proxy
- **Patterns de Comportamento (*behavioral*)**
  - Chain of Responsibility, Command, Interpreter, Iterator, Mediator, Memento, Observer, State, Strategy, Template Method, Visitor

Design Pattern Singleton

Propósito

Garantir que uma classe tenha somente uma instância, e fornecer um ponto de acesso global a ela.

Motivação

É importante para algumas classes ter exatamente uma instância. (...) Como nós garantimos que uma classe tem somente uma instância e que a instância seja facilmente acessível? (...) A melhor solução é tornar a própria classe responsável por esta única instância.

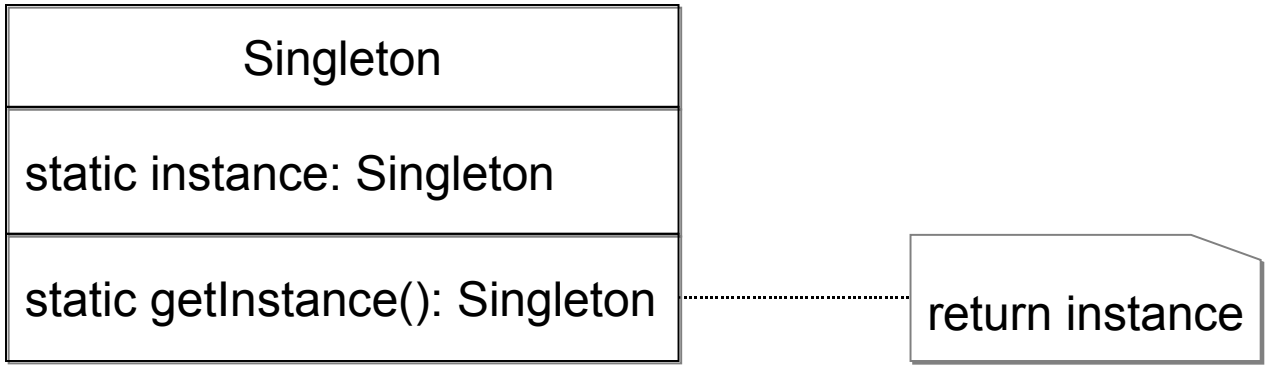
Design Pattern Singleton

Aplicabilidade

Use o pattern Singleton se deve existir exatamente uma instância de uma classe, e esta deve ser acessível de um ponto de acesso bem conhecido.

Design Pattern Singleton

Estrutura



Design Pattern Singleton

Participantes

Singleton:

define uma operação `getInstance()` que deixa os clientes acessarem sua única instância;

pode ser responsável por criar sua própria instância.

Colaborações:

clientes acessam uma instância de Singleton somente através da operação getInstance() de Singleton.

Design Pattern Singleton

Código-Exemplo

```
// Singleton.java
public final class Singleton {
  private static Singleton instance = null;

  private Singleton() { ... }

  public static Singleton getInstance() {
    if (instance==null) {
      instance = new Singleton();
    }
    return instance;
  }
}
```

Design Pattern Singleton

Código-Exemplo

```
public class UsoDoSingleton {
     :
   Singleton obj;
     :
   obj = Singleton.getInstance();
     :
}
```

alguns patterns conhecidos ...

Model-View-Controller (MVC) [2]

Proxy [1] [2]

Iterator [1]

Master-Slave [2]

Forwarder-Receiver [2]

resultados esperados ...

- formação de vocabulário comum para expressar conceitos e formas de relacioná-los
- criação de uma literatura que ajude o desenvolvedor de software a resolver problemas recorrentes
- criação de uma linguagem para a comunicação entre iniciantes e *experts* sobre estes problemas e suas soluções

Onde estão os patterns ?

1. **Design Patterns**

   Elements of Reusable Object-Oriented Software

   Erich Gamma

   Richard Helm

   Ralph Johnson

   John Vlissides

   Editora Addison Wesley

Onde estão os patterns ?

2. **Pattern-Oriented Software Architecture**
   A System of Patterns
   Frank Buschmann
   Regine Meunier
   Hans Rohnert
   Peter Sommerlad
   Michael Stal
   Editora John Wiley & Sons

Onde estão os patterns ?

3. **Pattern-Oriented Software Architecture Volume 2**
   Patterns for Concurrent and Networked Objects
   Douglas Schmidt
   Michael Stal
   Hans Rohnert
   Frank Buschman
   Editora John Wiley & Sons

Onde estão os patterns ?

4. **Pattern Languages of Program Design**
   Volumes 1, 2, 3, 4
   International Conference on Pattern Languages of Programming
   Editora Addison-Wesley

5. **The Hillside Group Web Pages**
   http://www.hillside.net

Onde estão os patterns ?

6. **Pattern Languages of Programs (PLoP)**

   http://jerry.cs.uiuc.edu/~plop

7. **Patterns and Software: Essential Concepts and Terminology**

   http://www.cmcrossroads.com/bradapp/docs/patterns-intro.html

Prof. Sérgio Teixeira de Carvalho
sergiocarvalho@anhanguera.edu.br

Slides
http://www.sergiocarvalho.net

Design Pattern para Configuração de Arquiteturas de Software
http://www.sergiocarvalho.net