Sérgio Taborda's Weblog

Alguns ensinam. Alguns fazem. O resto procura nos livros.

Type text to search here...

Separação de Responsabilidades e Encapasulamento

Deixe o seu comentário Go to comments

Descrever os axiomas fundamentais da orientação a objetos em poucas palavras é uma tarefa difícil. Mas, ao mesmo tempo, vital. Não é de imaginar que alguém possa usar uma linguagem orientada a objetos (OO) sem os conhecer. Não menos importante que os conhecer é saber identificá-los e usá-los.

O primeiro , e talvez o único, principio de OO é o da Separação de Responsabilidades. O termo em inglês *Separation of Concerns* (SoC) define melhor o significado. *Concerns* não é só a responsabilidade, mas também a preocupação: "Principio de Separação de Preocupações" dá uma idéia melhor do que se quer dizer. Em resumo, um objeto deve fazer apenas uma tarefa e fazê-la bem. Não se deve *preocupar* com o que os outros objetos fazem. Num sistema macroscópico em que um objeto coordena objetos menores, ele confia que esses objetos executam bem a sua tarefa e não se *preocupa* com o como eles a executam.A única coisa que tem que existir entre todos os objetos é o consenso de com o quê , cada um, se deve *preocupar*.

A preocupação de cada objeto é apenas zelar pela melhor concretização da sua responsabilidade no sistema e zelar para que nada corrompa o controlo que ele tem sobre como executar sua tarefa.
Mas como conseguir a separação de responsabilidades ? A única resposta a esta pergunta é: Encapsulamento. É através das diferentes formas de encapsulamento que o objeto concretiza seu objetivo.

1. Talvez a menos óbvia forma de encapsulamento é o próprio objeto em si. O fato de podermos juntar numa entidade só estado e comportamento é uma forma de encapsulamento.
2. Uma forma mais conhecida de encapsulamento é o nivel de oclusão. Ou seja, o nivel em que o objeto deixa visivel ,ou não, aos outros objetos aquilo que ele é, ou faz.

3. A herança é também um forma de encapsulamento. O fato de um objeto se esconder sobre uma ou mais capas de identidade esconde dos outros objetos seu verdadeiro funcionamento e intenções.

Em Java estas três formas de encapsulamento e algumas regras si em que o Principio da Separação de Responsabilidades seja semp efinição da linguagem define os três artefatos necessários.

1. A **Classe** detém todos os conhecime estruir, o que ele faz e o que é. A Classe é a tradução da prim
2. Os **modificadores de acesso** (defaul uem diferentes níveis de oclusão implementando assim, a seg
3. A **Interface** dá corpo à terceira form ue um só objeto tenha várias caras e execute o mesmo traba

Embora os artefatos embutidos na linguage construção de objetos e implantação do Principio de Separação d ntes combinações possíveis. O uso exaustivo dessas combina junto de padrões recorrentes que poderemos então usar diretamente na construção de nossos objetos: os Padrões de Projeto.

Comentários (0) Trackbacks (1) Deixe o seu comentário Trackback

1. Ainda sem comentários.

1. 2014/04/01 às 15:04
   Gerenciando o tempo de vida do Entity Framework DBContext | Eriko Morais

Deixar uma resposta

Escreva o seu comentário aqui...

RSS feed

Artigos recentes

- O movimento perpétuo e a energia eterna
- O fuso e a roca

- Voto Consciente

Blog no Java Buinding

Com a inauguração do JavaBuilding as minhas obervações sobre desenvolvimento de software em geral e sobre Java e Scrum em particular podem agora ser seguidas no meu novo blog Caderno Sérgio Taborda no JavaBuiliding. Este blog permance apenas para assuntos não relacionados a desenvolvimento de software.

Caderno no Javabuilding

- O Paradoxo do Inventor
- Coleções turbinadas
- Streams no Java 8 e em outras Linguagens
- Variância
- Java 8 – Prólogo
- Monads em Java
- Scala: O vencedor da batalha Java vs .Net

MiddleHeaven

- O caso de Enumerable infinito
- MiddleHeaven e Java 8
- Javadoc Disponivel
- Lista de Discussão
- Seis anos e muito para fazer
- Seguindo em frente
- No céu do meio
- Novo Conteudo
- Utilitários: Coleções aumentadas
- Nosso novo blog

Twitter

- O Paradoxo do Inventor - Como pensar grande dá mais resultado ow.ly/NiDAH 1 month ago
- Entenda mais sobre como a nova API de Stream vai mudar sua forma de programar e como ela afetou o design do java 8 ow.ly/LGYPG 2 months ago
- A variancia em java e outras linguagens ow.ly/Li320 2 months ago
- Monads em Java ow.ly/qs5Qo 1 year ago
- O vencedor da batalha Java vs .Net ow.ly/qcSCr 1 year ago

Meta

- Registar
- Iniciar sessão
- RSS dos artigos
- Feed RSS dos comentários.
- Create a free website or blog at WordPress.com.

Páginas

- Desenvolvimento de Software

    - A Arte de Fabricar Software
    - Arquitetura
        - Arquitetura Orientada ao Domínio
        - Arquitetura Web
    - Java
        - Coleções: Como não usar Arrays
        - Do DAO ao Domain Store
        - Exceções: Boas Práticas, Más Práticas
        - Exceções: Classes Utilitárias
        - Exceções: Conceitos
        - FAQ
            - Primeiro Programa
            - Sorteio aleatório sem Repetição
            - Trabalhando com Números
        - Igualdade em Java
        - Introspeção
        - java.lang.Object
        - OO
            - Herança
            - Polimorfismo
            - Separação de Responsabilidades e Encapasulamento
        - Os 10 mandamentos do bom programador Java
        - Palavras Reservadas
        - Patterns
            - Adapter
            - Bean
            - Builder
            - Composite
            - DAO
            - Factory
            - Factory Method
            - Fastlane
            - Iterator
            - Money
            - MoneyBag
            - MVC
            - Query Object
            - Repository
            - Singleton
            - Transfer Object
            - Value Object
    - Scrum
        - Equipe
        - Planejamento
        - Produto e Projeto
        - Projeções
        - Sprint
        - Valores
- Física
    - Mecânica Quântica
- Livros
- Magic: The Gathering
    - O Segredo do Magic
- Sobre mim

Topo