**Guia de Implementação**

SUMÁRIO

[1. Introdução 2](#_Toc485801465)

[1.1. Referências 2](#_Toc485801466)

[2. Organização e Estilo do Código 2](#_Toc485801467)

[3. Comentários 2](#_Toc485801468)

[4. Nomeação 2](#_Toc485801469)

[5. Declaração 3](#_Toc485801470)

[6. Reutilização 3](#_Toc485801471)

[7. Tratamento de Erros e de Exceções 3](#_Toc485801472)

[8. teste unitário 3](#_Toc485801473)

[9. Diretrizes Gerais 4](#_Toc485801474)

# Introdução

[A introdução deve fornecer uma visão geral do **Guia de Desenvolvimento**. Ela contém a finalidade, o escopo, as definições, os acrônimos, as abreviações, as referências e a visão geral deste documento.]

## Referências

[Esta subseção deve apresentar uma lista completa de todos os documentos mencionados no **Guia de Desenvolvimento**. Cada documento deve ser identificado por um título. Ex.:

* Guia - Oracle Java Code Conventions.pdf;
* Guia - Padroes e Ferramentas para Projeto de Banco de Dados.pdf ]

# Organização e Estilo do Código

[Uma descrição do tamanho e do layout de um módulo, do espaço, aninhamento, legibilidade, etc. Ex.:

* Tamanho de classe: não mais que 1000 linhas
* Tamanho demétodo: não mais que 20 linhas

]

# Comentários

[Uma descrição do uso de comentários.Ex.:

Java Doc de Classes

/\*\*

\* <Descrição da funcionalidade da Classe>

\* @author: <Fulano da Silva>

\*/

Java Doc de Métodos

/\*\*

\* <Descrição completa das funcionalidades do método>

\* @return <tipo do retorno do método: String, int, etc>

\* @param <parâmetro1> <Descrição do parâmetro>

\* @param <parâmetro2> <Descrição do parâmetro>

\* @exception <exceção retornada>

\*/

]

# Nomeação

[Uma descrição de como cada tipo de entidade (como, por exemplo, arquivos, tipos, variáveis e subprogramas) deverão ser nomeados. Ex.:

Arquivo Java

* NomeClasse.java

Scripts SQL

* NOMESCRIPT.sql

]

# Declaração

[Uma descrição de como as declarações devem ser escritas, no caso do padão prever mais de uma boa prática.]

1. Padrão de indentação de declaração de métodos:

someMethod(int anArg, Object anotherArg, String yetAnotherArg,

Object andStillAnother) {

...

}

]

# Reutilização

[Esta seção fornece regras e diretrizes para facilitar a reutilização. Ex.:

* Criar interfaces entre as camadas arquiteturais: Apresentação, Controle e Dados;
* Aplicar os padrões de atribuição de responsabilidade nas classes (ver padrões GRASP).

]

# Tratamento de Erros e de Exceções

[Esta seção fornece regras e diretrizes sobre como os erros e as exceções devem ser tratados. Ex.:

* Criar uma casse de tratamento de exceções na camada de apresentação.
* Garantir que as exceções subam para a camada de apresentação, usando a clausula Trows nos métodos envolvidos.

]

# teste unitário

[Esta seção deve descrever como os testes unitários devem ser desenvolvidos. Por exemplo, em Java deve-se usar o Framework de testes Junit. Ex.:

* Criar uma suite de testes com todos os métodos públicos da Facade.

]

# Diretrizes Gerais

[Esta seção deve descrever, caso existam, diretrizes e estratégias a serem usadas no desenvolvimento].