**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

Este Guia de Desenvolvimento tem como objetivo fornecer uma referência clara e abrangente para os desenvolvedores que trabalham no projeto SVSA. Ele delineia as diretrizes, padrões e práticas recomendadas que devem ser seguidos durante o processo de desenvolvimento, assegurando a qualidade e a consistência do código.

O SVSA é o Sistema de Vigilância Socioassistencial. Ele é um instrumento de gestão de informações que tem como objetivo analisar e disseminar dados sobre a situação de vulnerabilidade e risco social de famílias e indivíduos. O sistema já é existente, porém dentro desse projeto iremos desenvolver novas funcionalidades a ele.

O projeto é desenvolvido por Java JSF para backend, PrimeFaces para frontend e o MySQL para o banco de dados. Este guia tem como objetivo referenciar as convenções utilizadas e mencionar como o projeto deve ser estruturado de acordo com essas convenções.

## Referências

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

# Organização e Estilo do Código

* Arquivos maiores do que 2000 linhas são complicados e devem ser evitados
* Um arquivo consiste em seções que devem ser separadas por linhas em branco e um comentário opcional, identificando cada seção.
* Tamanho de classe: não mais que 1000 linhas
* Tamanho de método: não mais que 20 linhas

# Comentários

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

Arquivo Java (Pacote model):

* NomeDaClasse.java

Arquivo Java (Pacote controller):

* NomeDaClasseBean.java

Arquivo Java (Pacote service):

* NomeDaClasseService.java

Arquivo Java (Pacote DAO):

* NomeDaClasseDAO.java

Arquivos Java (Pacote Util):

* NomeDaClasseUtil.java

# Declaração

1. Padrão de indentação de declaração de métodos:

someMethod(int anArg, Object anotherArg, String yetAnotherArg,

Object andStillAnother) {

...

}

# Reutilização

* Aplicar os padrões de atribuição de responsabilidade nas classes (ver padrões GRASP).

# Tratamento de Erros e de Exceções

* Garantir que as exceções subam para a camada de apresentação, usando a clausula Trows nos métodos envolvidos.

# teste unitário

# Diretrizes Gerais

* Padrões de Codificação: Adotar convenções de nomenclatura e estrutura de código consistentes com as diretrizes do projeto SVSA.
* Controle de Versão: Utilizar o sistema de controle de versão para registrar alterações e facilitar a colaboração entre os desenvolvedores, seguindo as práticas do projeto SVSA.
* Revisão de Código: Realizar revisões de código regulares para promover a qualidade e a troca de conhecimento entre a equipe.[[1]](#footnote-7074)

1. [↑](#footnote-ref-7074)