****

**Serviço Nacional de Aprendizagem Industrial**

**SENAI “Gaspar Ricardo Junior”**

Curso

TÉCNICO EM DESENVOLVIMENTO

DE SISTEMAS

Métodos equals e hashCode em Java e o uso de Lombok para otimizar código em ambientes de desenvolvimento.

Nathalia Almeida Yoshioka

Sorocaba

Novembro – 2024



**Serviço Nacional de Aprendizagem Industrial**

**SENAI “Gaspar Ricardo Junior”**

Nathalia Almeida Yoshioka

Métodos equals e hashCode em Java e o uso de Lombok para otimizar código em ambientes de desenvolvimento.

Este trabalho investiga a importância e o uso dos métodos equals e hashCode em Java.

Prof. – Emerson Magalhães

Sorocaba

Novembro – 2024

HISTÓRICO DE VERSÕES

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Sumário**

[RESUMO 4](#_Toc158208589)

[OBJETIVO 5](#_Toc158208590)

[INTRODUÇÃO 6](#_Toc158208591)

[1. TÍTULO 1 7](#_Toc158208592)

[1.1. SUBTOPICO 1 7](#_Toc158208593)

[1.2. SUBTOPICO 2 7](#_Toc158208594)

[2. TÍTULO 2 8](#_Toc158208595)

[2.1. SUBTOPICO 1 8](#_Toc158208596)

[2.2. SUBTOPICO 2 8](#_Toc158208597)

[2.3. SUBTOPICO 2 - NÍVEL 1 8](#_Toc158208598)

[2.3.1. SUBTOPICO 2 – NÍVEL 2 – TEMA 1 8](#_Toc158208599)

[2.3.2. SUBTOPICO 2 – NÍVEL 2 - TEMA 2 9](#_Toc158208600)

[CONCLUSÃO 10](#_Toc158208601)

[BIBLIOGRAFIA 11](#_Toc158208602)

[LISTA DE FIGURAS 12](#_Toc158208603)

[LISTA DE TABELAS 12](#_Toc158208604)

# 

Tema do Trabalho

# RESUMO

O trabalho de pesquisa tem como objetivo explorar a importância e o funcionamento dos métodos **equals** e **hashCode** em Java, analisando como influenciam o comportamento de coleções baseadas em hashing e a gestão de entidades em frameworks como o Spring. O estudo detalha a necessidade de uma implementação correta desses métodos para garantir a integridade e a eficiência das operações com objetos em coleções como **HashMap** e **HashSet.**

# OBJETIVO

O objetivo desta pesquisa é explorar a importância e o funcionamento dos métodos equals e hashCode em Java, analisando como eles influenciam o comportamento de coleções que utilizam hashing e como são usados para gerenciar entidades em frameworks como o Spring. Além disso, a pesquisa visa investigar como a biblioteca Lombok pode simplificar a implementação desses métodos e otimizar o desenvolvimento de software em Java.

# INTRODUÇÃO

Na programação orientada a objetos em Java, a correta implementação dos métodos equals e hashCode é essencial para garantir a integridade e eficiência das operações com objetos. Estes métodos são particularmente importantes em coleções baseadas em hashing, como HashMap e HashSet, onde desempenham um papel crucial na comparação e armazenamento de objetos.

Além das coleções, a implementação de equals e hashCode é vital em frameworks populares como o Spring, onde são frequentemente utilizados para gerenciar entidades, persistência e operações de caching. A implementação manual desses métodos pode ser repetitiva e propensa a erros, o que destaca a importância de ferramentas que possam automatizar esse processo.

A biblioteca Lombok emerge como uma solução eficiente para reduzir o código boilerplate associado à implementação desses métodos. Com anotações como @EqualsAndHashCode, Lombok facilita a geração automática de equals e hashCode, melhorando a legibilidade e manutenção do código. No entanto, o uso de Lombok também levanta questões sobre dependência de bibliotecas externas e possíveis desafios de depuração.

Esta pesquisa busca oferecer uma visão abrangente sobre a importância de equals e hashCode, os impactos no comportamento das coleções Java e a utilização prática em frameworks. Também investigará como Lombok pode simplificar esses processos, apresentando vantagens e desvantagens de sua adoção em projetos de desenvolvimento Java.

# CONTEXTUAÇIZAÇÃO DOS MÉTODOS EQUALS E HASHCODE

Os métodos **equals** e **hashCode** são fundamentais na programação em Java, especialmente no contexto de coleções e gerenciamento de entidades. Esses métodos são utilizados para comparar objetos e determinar a igualdade entre eles, desempenhando um papel crucial em estruturas de dados baseadas em hashing como **HashMap** e **HashSet.**

## HashMap

HashMap é uma implementação da interface Map em Java, que armazena pares chave-valor. Este tipo de coleção utiliza uma tabela de dispersão (hash table) para garantir uma busca rápida de valores com base em suas chaves. Algumas das principais características do HashMap são:

* Complexidade de Acesso: O acesso a valores com base nas chaves geralmente é O①, o que significa que é muito rápido.
* Permite Nulos: Pode conter uma chave nula e vários valores nulos.
* Não Ordenado: Os elementos não têm uma ordem específica.

## HashSet

HashSet é uma implementação da interface Set em Java, que armazena apenas elementos únicos e não permite duplicatas. Ele também é baseado em uma tabela de dispersão para garantir uma verificação eficiente de duplicatas. Algumas das principais características do HashSet são:

* Complexidade de Acesso: A verificação da presença de elementos geralmente é O①.
* Não Permite Nulos Duplicados: Pode conter um único valor nulo, mas não permite valores duplicados.
* Não Ordenado: Os elementos não têm uma ordem específica.

# IMPORTÂNCIA DOS MÉTODOS PARA COLEÇÕES E FRAMEWORKS

Implementar corretamente equals e hashCode garante que coleções que utilizam hashing funcionem adequadamente, evitando problemas de duplicidade e garantindo a eficiência das operações. Em frameworks como Spring, esses métodos são essenciais para o gerenciamento de entidades, persistência e caching.

# INTRODUÇÃO AO LOMBOK E SUA FINALIDADE NO DESENVOLVIMENTO EM JAVA

Lombok é uma biblioteca que visa reduzir o código boilerplate em Java, facilitando a implementação de métodos comuns como equals e hashCode através de anotações, melhorando a produtividade e legibilidade do código.

# FUNDAMENTOS TEÓRICOS

## Explicação do Contrato entre equals e hashCode

O contrato entre equals e hashCode define regras que devem ser seguidas para garantir a correta funcionalidade de objetos em coleções baseadas em hashing:

* Se dois objetos são iguais (equals retorna true), então eles devem ter o mesmo valor de hashCode.
* Se dois objetos têm o mesmo valor de hashCode, eles não necessariamente são iguais.
* Consistência: o valor de equals e hashCode não deve mudar enquanto os objetos comparados não forem modificados.

## Como o Contrato entre equals e hashCode Afeta o Comportamento das Coleções

Coleções como HashMap e HashSet utilizam os métodos hashCode para armazenar objetos de maneira eficiente. Uma implementação inadequada pode resultar em perda de desempenho e comportamentos inesperados, como a incapacidade de recuperar corretamente os elementos armazenados.

## Importância da Implementação Correta de equals e hashCode em Entidades de Aplicações Java

A correta implementação desses métodos é crucial para garantir que as entidades sejam corretamente gerenciadas em operações de persistência, caching e comparação dentro de aplicações empresariais.

# UTILIZAÇÃO PRÁTICA EM COLEÇÕES JAVA E NO SPRING

## Exemplos Práticos em Coleções como HashSet e HashMap



Figura 1 - Exemplo de coleções HashMap e HashSet

## Exemplo Prático de uma Entidade Spring



Figura 2 - Exemplo de Entidade Spring

# LOMBOK

Lombok é uma biblioteca Java que reduz significativamente o código boilerplate ao permitir que os desenvolvedores anotem suas classes com anotações específicas, que geram automaticamente métodos como equals e hashCode.

Para boas práticas, é importante garantir que a equipe esteja familiarizada com Lombok e que existam processos de revisão de código para monitorar o uso adequado das anotações. Além disso, deve-se avaliar a necessidade de Lombok em projetos onde a simplicidade do código e a facilidade de depuração são prioridades.

## Análise das Anotações @EqualsAndHashCode e @Data



Figura 3 - Exemplo de anotation DATA

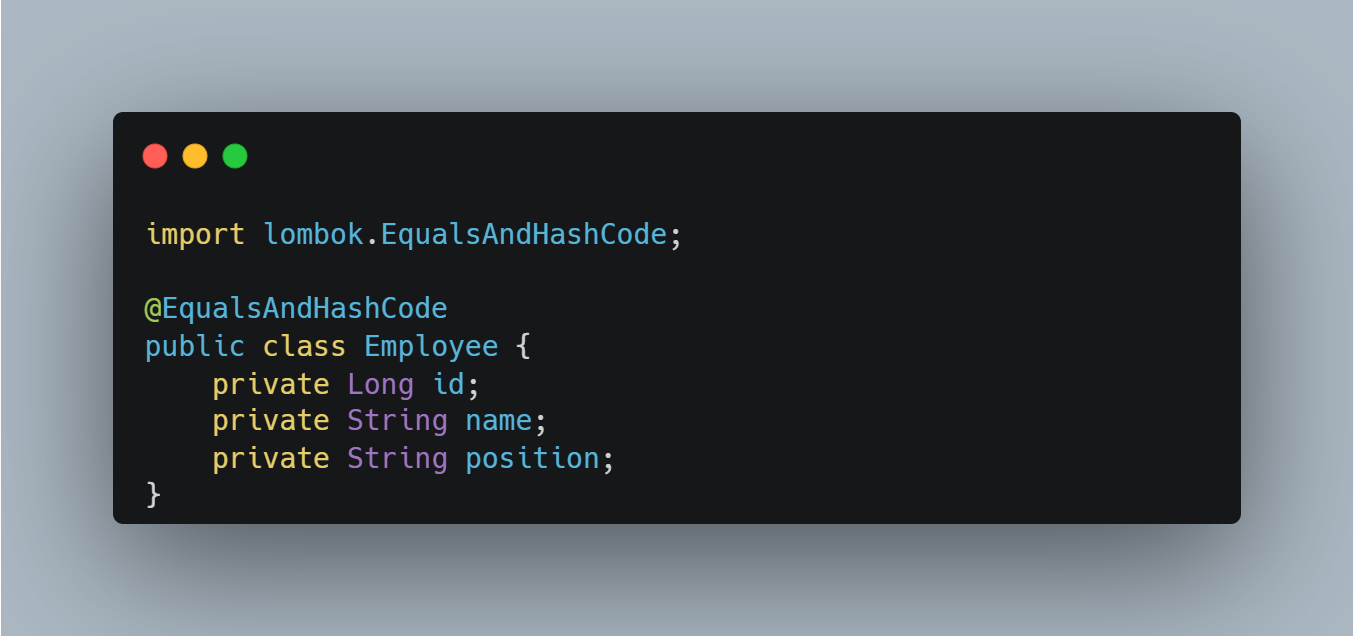


Figura 4 - Exemplo de anotation EqualsAndHashCode

## Vantagens e Desvantagens de Usar Lombok para equals e hashCode

**Vantagens:**

* **Redução de Código Boilerplate:** Lombok elimina a necessidade de escrever manualmente métodos repetitivos.
* **Melhor Legibilidade e Manutenção do Código:** O código fica mais limpo e fácil de manter.

**Desvantagens:**

* **Dependência de uma Biblioteca Externa:** Pode haver complicações se houver necessidade de remover a biblioteca no futuro.
* **Potenciais Problemas com Depuração de Código (Debugging) e Geração de Código com Lombok:** A geração de código automática pode dificultar a depuração.

# CONCLUSÃO

A implementação correta dos métodos equals e hashCode é essencial para o funcionamento adequado de coleções e frameworks como Spring. A biblioteca Lombok simplifica esse processo, reduzindo o código boilerplate e melhorando a legibilidade. No entanto, é importante considerar as desvantagens e garantir que o uso de Lombok seja bem gerenciado em ambientes de produção. Com esses conhecimentos, desenvolvedores podem criar aplicações Java mais eficientes e escaláveis.

# BIBLIOGRAFIA

**OBJECT (JAVA PLATFORM SE 8).** Disponível em:< <https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html#equals-java.lang.Object->>. Acesso em: 11 nov. 2024.

**@EQUALSANDHASHCODE.** Disponível em:< <https://projectlombok.org/features/EqualsAndHashCode>>. Acesso em: 11 nov. 2024.

DINAKARAN RAMADURAI. **Pitfalls of Lombok — Efficient way to use EqualsAndHashCode in the entity class**. Disponível em:<<https://dinakaranramadurai.medium.com/pitfalls-of-lombok-efficient-way-to-use-equalsandhashcode-in-the-entity-class-d06d8b9dbc0b>>. Acesso em: 11 nov. 2024.

**Project Lombok**. Disponível em: <<https://projectlombok.org/>>.

‌

# LISTA DE FIGURAS

Figura 1 – Exemplo de coleções HashMap e HashSet, 2024. Criada com < <https://carbon.now.sh/>>.

Figura 2 – Exemplo de Entidade Spring, 2024. Criada com < <https://carbon.now.sh/>>.

Figura 3 – Exemplo de anotation DATA, 2024. Criada com < <https://carbon.now.sh/>>.

Figura 4 – Exemplo de anotation EqualsAndHashMap, 2024. Criada com < <https://carbon.now.sh/>>.