



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**

Luany Urtado Santos

Sorocaba
Novembro – 2024



SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL

SENAI “GASPAR RICARDO JUNIOR”

Luany Urtado Santos

MÉTODOS EQUALS E HASHCODE EM JAVA E O USO DE LOMBOK

Breve pesquisa sobre os métodos equals e hashCode e o uso
do lombok para otimizar código em ambientes de
desenvolvimento

Prof. – Emerson Magalhães

Sorocaba
Novembro – 2024

INTRODUÇÃO

Os métodos `Equals` e `HashCode` são recursos da linguagem da programação Java que servem para comparar, ordenar e distinguir objetos.

Equals: compara dois objetos e retorna um booleano, indicando se são iguais ou não.

HashCode: gera um número que é usado para organizar e agrupar objetos em listas, de modo a organizá-los mais rapidamente.

Eles são essenciais na gestão de entidades, especialmente em operações de persistência e caching.

O Lombok é uma biblioteca Java focada em produtividade e redução de código boilerplate que, ajuda a facilitar a criação de getters, setters, construtores e métodos como os que estão sendo tratados agora (`equals` e `hashCode`).

FUNDAMENTOS TEÓRICOS

No `Equals` ele estabelece que dois objetos são considerados iguais,

tendo que retornar o valor no método `hashCode`. Sua implementação deve ser transitiva, simétrica, consistente e considerando objetos nulos.

O contrato garante que objetos iguais tenham o mesmo `hashCode`, isso significa que é crucial para o funcionamento de coleções como o `HashSet` e o `HashMap` que usam o `hashCode` para localizar objetos. Com isso, o má funcionamento pode causar a falha ao buscar, inserir ou remover itens corretamente.

UTILIZAÇÃO PRÁTICA EM COLEÇÕES JAVA E SPRING

```
import java.util.HashSet;
import java.util.Set;

public class TesteHashSet {
    public static void main(String[] args) {
        Set<Pessoa> pessoas = new HashSet<>();

        // Criando algumas pessoas
        Pessoa p1 = new Pessoa("João", "joao@example.com");
        Pessoa p2 = new Pessoa("Maria", "maria@example.com");
        Pessoa p3 = new Pessoa("João", "joao@example.com"); // Mesmo nome e email que p1

        // Adicionando as pessoas no HashSet
        pessoas.add(p1);
        pessoas.add(p2);
        pessoas.add(p3); // Esta pessoa é "igual" a p1, então não será adicionada

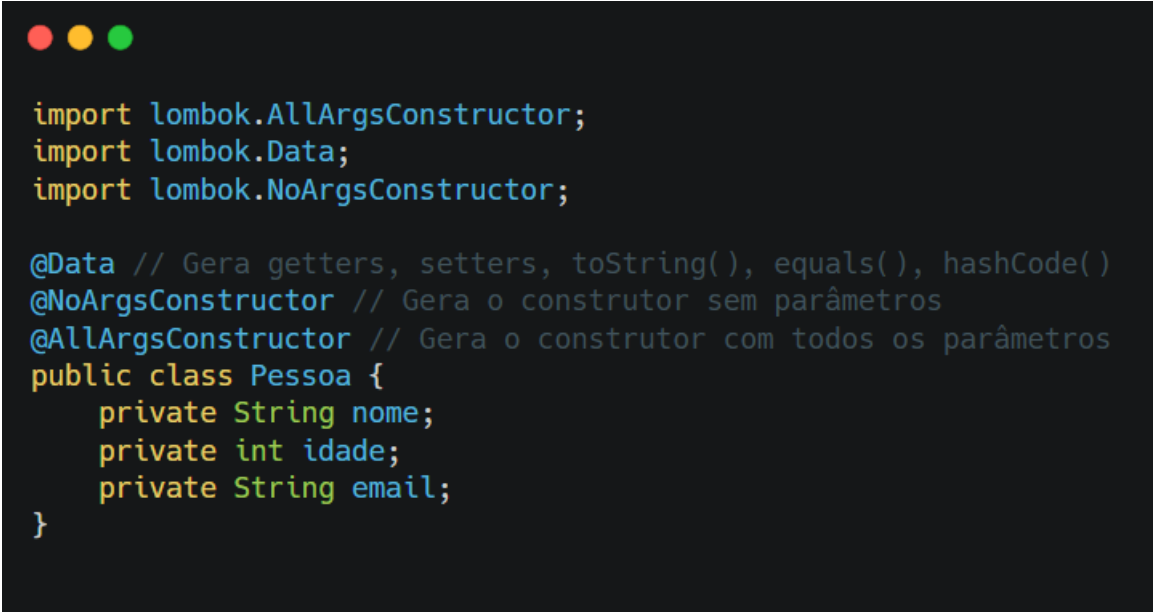
        // Exibindo as pessoas no HashSet
        System.out.println(pessoas);
    }
}
```

LOMBOK: SIMPLIFICAÇÃO DO CÓDIGO

Vantagens: menos código repetitivo e facilitação na manutenção.

Desvantagens: dependência na biblioteca e possíveis dificuldades na depuração (“limpeza”).

Ao analisarmos a anotação `@EqualsAndHashCode` e `@Data`, percebemos que ele gera métodos necessários automaticamente com base nos campos da classe.



```
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data // Gera getters, setters, toString(), equals(), hashCode()
@NoArgsConstructor // Gera o construtor sem parâmetros
@AllArgsConstructor // Gera o construtor com todos os parâmetros
public class Pessoa {
    private String nome;
    private int idade;
    private String email;
}
```

VANTAGENS

Redução de código boilerplate e melhor legibilidade e manutenção do código.

DESVANTAGENS

Dependência de uma biblioteca externa e potenciais problemas com depuração e geração de código com Lombok.

CONCLUSÃO

A utilização correta de sua implementação é importante para a integridade das coleções em Java. A dependência do Lombok pode ser vista como uma desvantagem em alguns contextos, que são cruciais para desenvolvedores que buscam criar aplicações eficientes e escaláveis.