HashSet

Arthur Henrique Gonçalves
Pedro Henrique Costa Dias
Vitor Alexandre Hasselmann de Oliveira

Interface Set

Interface que faz parte do pacote “java.útil”

Representa grupos de elementos sem duplicatas

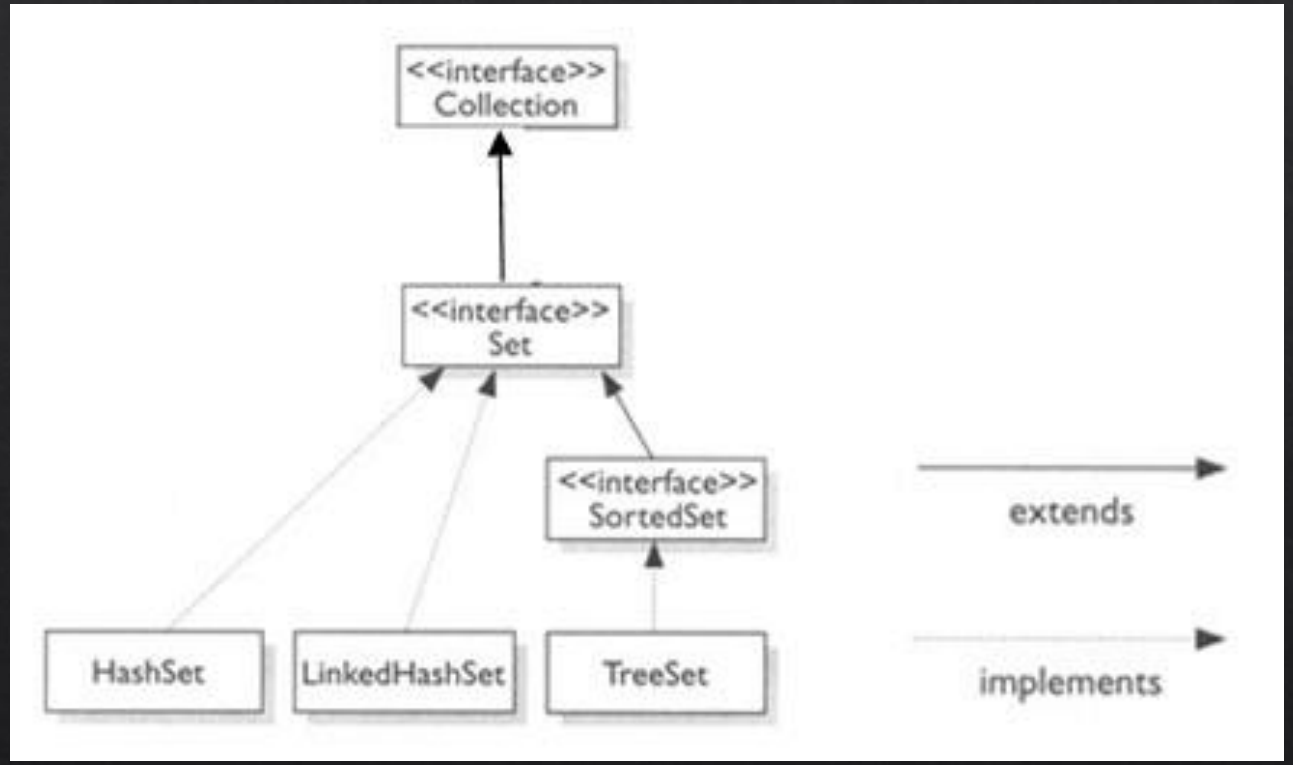
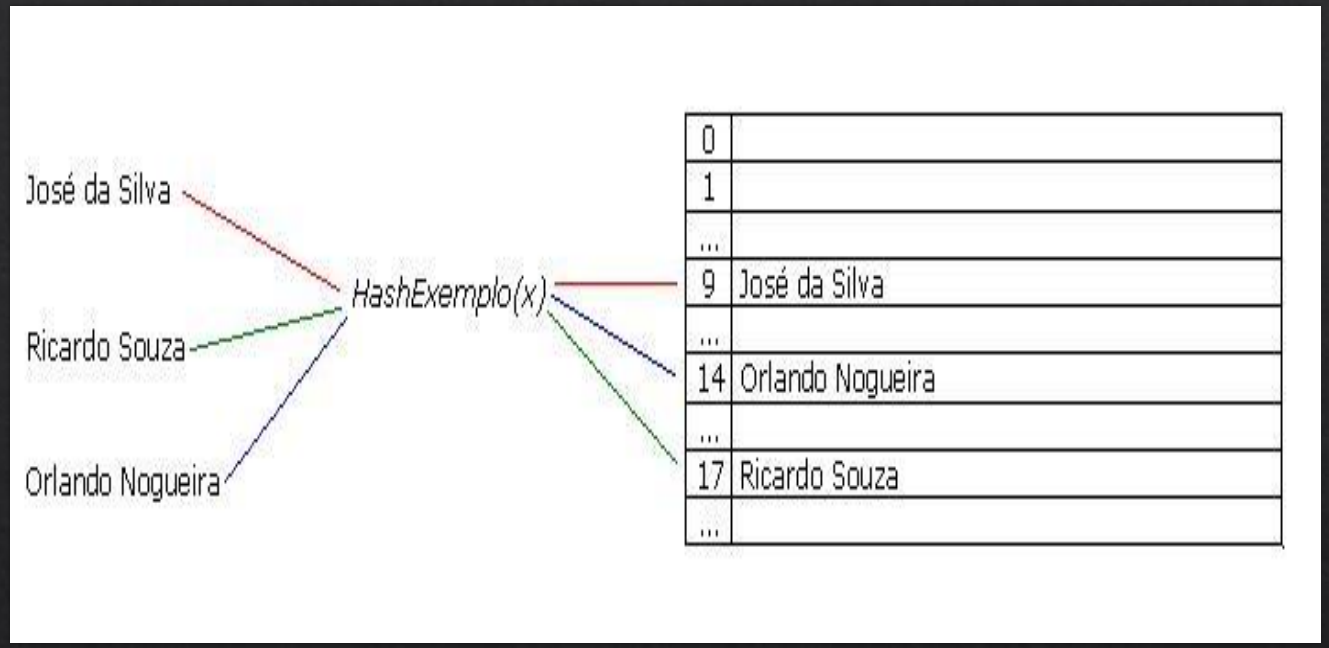


Tabela Hash

Tabela Hash = Tabela de dispersão

Estrutura de dados especial

Uso de chaves simples



É um tipo de tabela conhecida no ramo de ciência de computação, com o objetivo de facilitar a busca de um dado específico.
Utilizando-se de uma chave simples para tal busca.

HashSet

- O que é HashSet?

Uma coleção que armazena elementos únicos sem ordem específica.

Ideal para verificar presença de elementos, e realizar operações matemáticas em conjunto.

Características Essenciais

- Armazena elementos únicos: evita duplicatas e garante unicidade dos dados.
- Sem ordem específica: a ordem de inserção não é preservada, otimizando a pesquisa e a inserção.
- Eficiente: o uso de hashcode garante operações rápida de busca, adição e remoção.
- Implementação do Set: implementa a interface Set, fornecendo métodos específicos para conjuntos

Uso Adequado

- Verifica se um elemento existe em um conjunto
- Realiza operações matemáticas em conjunto (adição, multiplicação, subtração, e divisão)
- Armazena dados sem se preocupar com a ordem.

Implementação

- Implementação padrão: Java.util.HashSet (baseado em HashMap)

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

- Outras implementações: LinkedHashSet e TreeSet.

```java
import java.util.HashSet;
no usages
public class Main {
    no usages
    public static void main (String[] args){

// Criando um HashSet
        HashSet<String> nomes = new HashSet<>();

// Adicionando elementos
        nomes.add("Ana");
        nomes.add("João");
        nomes.add("Maria");
        nomes.add("Pedro");

// Verificando a presença de um elemento
        if (nomes.contains("Ana")) {
            System.out.println("Ana está presente!");
        }

// Removendo um elemento
        nomes.remove( o: "Pedro");

// Imprimindo o tamanho do HashSet
        System.out.println("Tamanho: " + nomes.size());
    }
}
```

Boas práticas e Dicas relevantes

- Evite usar objetos mutáveis como chaves: Pode levar a comportamento inesperados.
- Considere a ordem de inserção.
- Defina o tamanho inicial: Otimiza o desempenho para grandes conjuntos
- Utilize um método contains antes de adicionar: Evita duplicatas desnecessárias.

Exércicios para prática

- 1- Crie um HashSet de strings e adicione algumas palavras. Em seguida, imprima o conjunto para ver se as palavras foram adicionadas corretamente.

- 2- Crie um HashSet de números inteiros e adicione alguns números. Em seguida, imprima o conjunto para ver se os números foram adicionados corretamente.

- 3- Crie dois HashSets de strings e verifique se tem um elemento em comum usando o método retainAll()