**Anotações sobre collections**

Collection é um objeto que agrupa múltiplos elementos dentro de uma única unidade. Elas servem para armazenar e processar conjuntos de dados de forma eficiente. Antes do Java a implementação de coleções na linguagem incluía poucas classes e não tinha organização de um framework.

* Interfaces

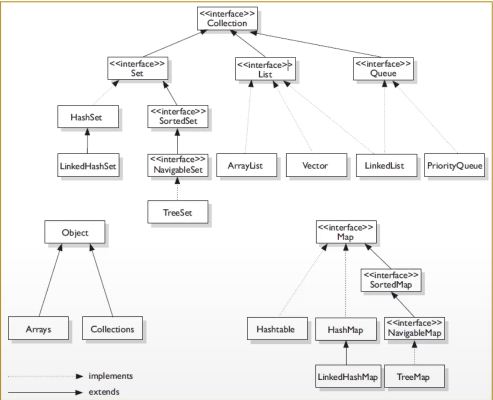
É um contrato que quando assumido por uma classe deve ser implementado;

* Implementações ou classes

São materializações a codificação das interfaces

* Algoritmos

Sequencia lógica, finita e definida de instruções que devem ser seguidas para resolver um problema.



**Java.until.list**

Elementos duplicados e garante ordem de inserção

O ArrayList e o vetor implementam só a interface List

A LinkedList e o PriorityQueue implementam a interface Queue

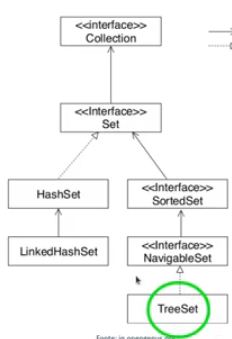
O arrayList é usado quando mias operações de pesquisa são necessárias, e o linkeList deve ser usado onde mais operações de inserção e exclusão são necessárias.

Interface gráfica do usuário, Texto

Descrição gerada automaticamente com confiança média

**Java.until.Set**

* Não permite elementos duplicados
* Não possui índices (não usa o indexOf,get ou set)
* Usando o set, você não vai conseguir exibir um elemento em uma posição especifica, nem adicionar e nem substituir um elemento. (só no lista da pra fazer)



* Hashset

é o mais rápido de todos, este usa HashTable e seus elementos não são ordenados, a complexidade desta estrutura é O(1), em outras palavras, não importa o quanto você adicione, remova, retire, o tempo de execução sempre será o mesmo.(admite 1 elemento null).

* LinkedHashSet

é derivada de HashSet, mas mantém uma lista duplamente ligada através de seus itens. Seus elementos são iterados na ordem em que foram inseridos. Opcionalmente é possível criar um LinkedHashSet que seja percorrido na ordem em que os elementos foram acessados na última iteração.(admite 1 elemento null)

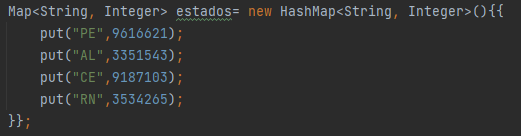
* TreeSet

implementa um algoritmo conhecido por red-black tree ou árvore rubro-negra. Sua principal característica é que ele é o único Set que implementa a interface SortedSet em vez de Set diretamente, mas de qualquer forma SortedSet implementa Set, assim continuamos tendo os mesmo métodos no **TreeSet**.(não admite elementos null).

**Map**

* as chaves são de valores únicos
* não possui uma forma de mostrar 1 item em uma posição específica.

Ordenação de Map



**Stream**

Reference Method

É um novo recurso do java 8 que permite fazer referência a um método ou construtor de uma classe e assim indicar que ele deve ser utilizado num ponto específico do código o deixando mais simples e legível. Para utilizá-lo, basta informar uma classe ou referência seguida do símbolo”:” e o nome do método sem os parênteses no final.





A Streams API traz uma nova opção para a manipulação de coleções em Java seguindo os princípios da programação funcional. Combinada com as expressões lambda, ela proporciona uma forma diferente de lidar com conjuntos de elementos, oferecendo ao desenvolvedor uma maneira simples e concisa de escrever código que resulta em facilidade de manutenção e paralelização sem efeitos indesejados em tempo de execução.

**Classe anônima**

É uma classe que não recebeu um nome e é declarado e instanciado em uma única instrução. O uso de uma classe anônima é necessário para criar uma classe que será instanciada apenas uma vez.

**Functional Interface**

É uma interface funcional e sua implementação pode ser tartada como expressões lambda.

**Lambda**

Uma função lambda é uma função sem declaração, isto é, não é necessário colocar nome, um tipo de retorno e o modificador de acesso. A ideia é que o método seja declarado no mesmo lugar em que será. As funções lambdas em Java tem sintaxe definida como (argumentos) ->(corpo).

**Operações intermediarias**

Operações que retornam uma stream encadeada fazem varias operações intermediarias uma atrás da outra.

**Operações Terminais**

Só podemos utilizar uma operação