INSTITUTO FEDERAL
PARANÁ

Orientação a Objetos II

Java Collections

Prof. Humberto Beneduzzi

Coleções genéricas

INSTITUTO FEDERAL
PARANÁ

- Uma coleção (*collection*) é um objeto que agrupa vários objetos.
- As coleções são utilizadas para armazenar e manipular dados.
- As estruturas de coleções do Java (***Framework Collections***) são estruturas de dados predefinidas, com interfaces e métodos para manipular esses dados.

Coleções genéricas

INSTITUTO FEDERAL
PARANÁ

- A ***Framework Collections*** contém:
  - **Interfaces** – são tipos de dados abstratos. As Interfaces permitem que as coleções sejam manipuladas independentemente dos detalhes das suas representações;
  - **Implementações** – São implementações concretas das coleções.

Coleções genéricas

INSTITUTO FEDERAL
PARANÁ

- **Algoritmos** – realizam operações sobre os elementos das coleções. Os algoritmos são polimórficos, ou seja, o mesmo método por ser utilizado em diferentes implementações de uma interface.

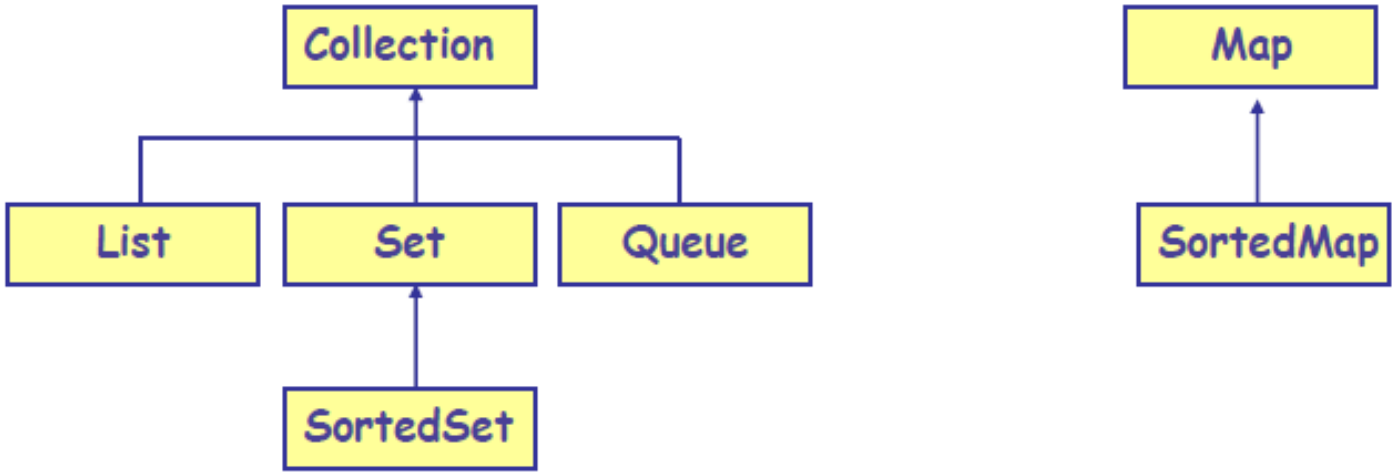
- **`- java.util`** é o package que contém o *Framework Collections*.

Coleções genéricas

INSTITUTO FEDERAL
PARANÁ

- Uma coleção é uma estrutura de dados, na realidade, um objeto que pode armazenar referências a outros objetos.
- Normalmente, as coleções contêm referências a objetos que são inteiramente do mesmo tipo.
- A seguir algumas interfaces da estrutura de coleções:

Coleções genéricas

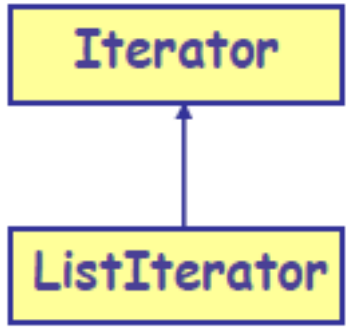


- **`Collection`**: interface-raiz, o Java não tem nenhuma implementação direta desta interface
- **`List`**: coleção ordenada que pode conter elementos duplicados;
- **`Set`**: coleção que não contém duplicatas;
- **`Queue`**: Em geral, interface de uma fila;
- **`Map`**: Associa chaves a valores e não pode conter chaves duplicadas.

Iterador

INSTITUTO FEDERAL
PARANÁ

- Com uma coleção de objetos, é muitas vezes necessário percorrer todos os objetos para realizar uma determinada ação em cada objeto.
- Um **iterador** permite percorrer uma coleção de objetos.
- Existem duas interfaces na plataforma Java que representam o conceito de iterador de uma coleção: **`Iterator`** e **`ListIterator`**.



Iterador

INSTITUTO FEDERAL
PARANÁ

- Um objeto do tipo **`Iterator`** tem os seguintes métodos:
  - **`hasNext`** – retorna **`true`** se a iteração tem mais elementos
  - **`next`** – retorna o próximo elemento na iteração
  - **`remove`** – remove o último elemento que foi retornado pelo método **`next`**.

Interface `List`

- A interface **`List`** representa uma coleção sequencial.
- As listas podem conter elementos duplicados.
- As implementações desta interface permitem que o utilizador tenha controle sobre a posição onde quer inserir, remover ou acessar elementos.

Interface `List`

INSTITUTO FEDERAL
PARANÁ

- A plataforma Java contém 3 implementações desta interface:
  - **`Vector`** é uma implementação desta interface através de um array que cresce dinamicamente à medida que lhe são inseridos elementos.
  - **`ArrayList`** e **`Vector`** têm comportamentos quase idênticos. A principal diferença é que **`Vector`** é sincronizado e **`ArrayList`** não é.
  - **`LinkedList`** é uma lista duplamente ligada.

Classe `ArrayList`

INSTITUTO FEDERAL
PARANÁ

- A classe de coleção **ArrayList <E>** (do pacote **java.util.ArrayList**) permite armazenar uma sequencia de objetos alterando dinamicamente seu tamanho.
- O **<E>** é um espaço reservado, que deve ser substituído pelo tipo de elementos que se deseja que o **ArrayList** contenha, no momento de sua declaração.

```
ArrayList <String> list = new ArrayList<String>();
```

Métodos da Classe `ArrayList`

INSTITUTO FEDERAL
PARANÁ

Método	Descrição
`boolean add(E e)`	Adiciona um elemento ao fim do ArrayList.
`void add(int index, E element)`	Insere o elemento especificado na posição especificada.
`boolean addAll(Collection<E> c)`	Adiciona todos os elementos de uma coleção existente no ArrayList.
`E remove(int index)`	Remove o elemento na posição especificada.
`boolean remove(Objetc o)`	Remove a primeira ocorrencia do elemento especificado.
`E set(int index, E element)`	Altera o elemento da posição especificada pelo elemento especificado.
`E get(int index)`	Retorna o elemento na posição especificada.

Métodos da Classe `ArrayList`

INSTITUTO FEDERAL
PARANÁ

Método	Descrição
`void clear()`	Remove todos os elementos do ArrayList
`Object clone()`	Retorna uma cópia independente do ArrayList.
`boolean contains(Object elemento)`	Retorna true se ArrayList contiver o elemento especificado; caso contrário false
`int indexOf(Object elemento)`	Retorna o índice da primeira ocorrencia do elemento especificado em ArrayList.
`void trimToSize()`	Reduz a capacidade de ArrayList de acordo com o número de elementos atual.
`boolean isEmpty()`	Retorna true se o ArrayList estiver vazio.
`int size()`	Retorna o número de elementos armazenados no ArrayList.