Linguagem de Programação Orientada a Objetos I

Coleções de Objetos - Lists

Prof. Tales Bitelo Viegas
https://fb.com/ProfessorTalesViegas

Introdução

- Coleção
  - É um objeto que representa um grupo de objetos, em outras palavras, é simplesmente um objeto que agrupa múltiplos elementos em uma simples unidade
  - Coleções são utilizadas para armazenar, recuperar, manipular, e comunicar dados agregados
  - Exemplos: baralho (uma coleção de cartas), caixa de correio (uma coleção de cartas), lista telefônica (uma coleção de nomes e telefones)

Introdução

- Implementação das Coleções
  - Três interfaces básicas na organização das coleções:
    - Conjuntos (sets)
    - Listas (Lists)
    - Mapas (Maps)

		Implementations				
		Hash Table	Resizable Array	Balanced Tree	Linked List	Hash Table + Linked List
Interfaces	Set	HashSet		TreeSet		LinkedHashSet
	List		ArrayList		LinkedList	
	Map	HashMap		TreeMap		LinkedHashMap

Lista de Objetos (List)

- Lista de Objetos (List)
  - Podem ser associadas como Arrays com algumas capacidades adicionais, uma das quais é a capacidade de ter seu tamanho modificado de acordo com a necessidade
  - Existe uma interface **List** que declara que métodos podem ser utilizados para manipulação de listas e duas classes que implementam esta interface

Lista de Objetos (List)

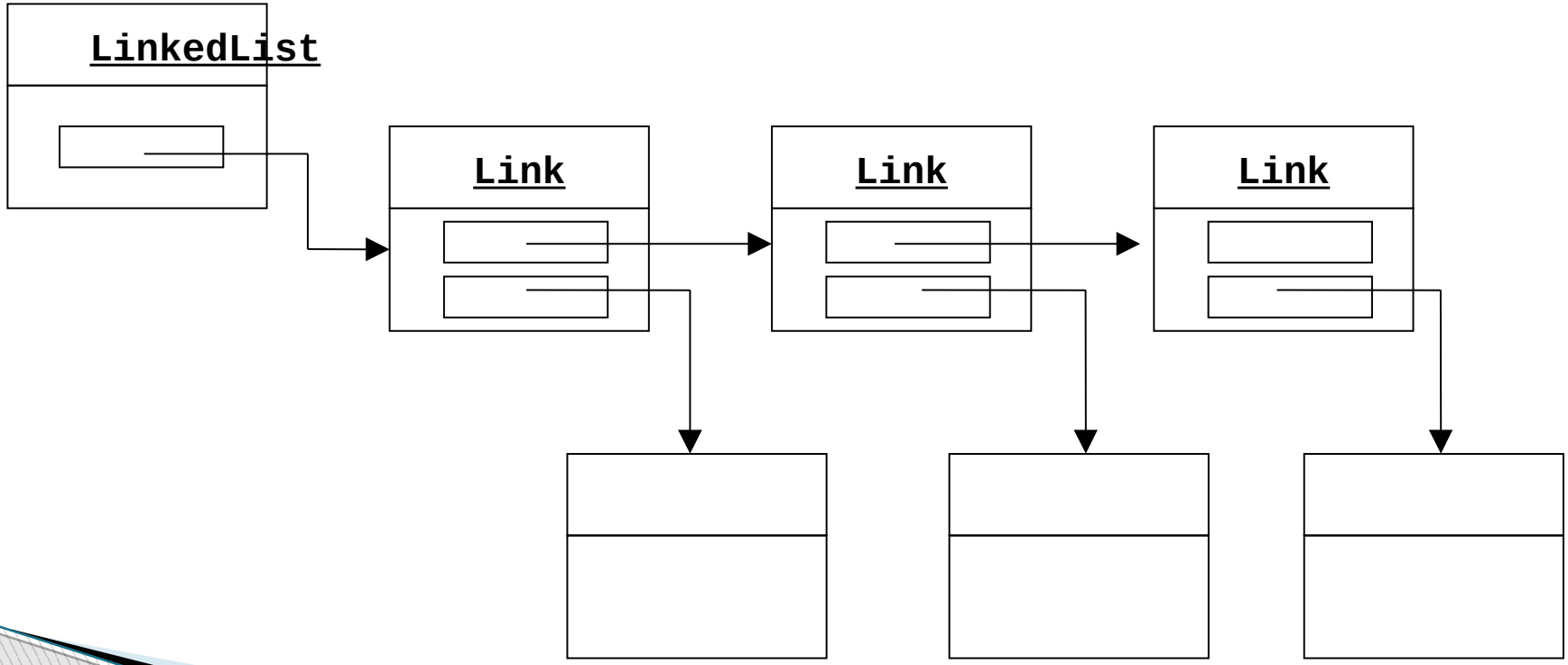
- Lista de Objetos (List)
  - Cada classe possui um mecanismo diferente para a representação interna dos objetos nas listas
  - Classes que implementam List
    - ArrayList
    - LinkedList
  - Pacote *java.util*
    - `import java.util.ArrayList;`
    - `import java.util.LinkedList;`
    - Consulte:
      - https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html
      - https://docs.oracle.com/javase/8/docs/api/java/util/LinkedList.html

Lista de Objetos (List)

- Lista de Objetos (List)
  - LinkedList
    - Têm melhor desempenho para as operações de inserção e remoção mas é, em geral, mais lenta para acesso sequencial aos elementos da lista
    - É mais conveniente para implementar pilhas e filas (métodos addFirst, addLast, getFirst, getLast, removeFisrt, removeLast)
  - ArrayList
    - Implementa a lista internamente como um array e tem desempenho melhor, exceto por operações como inserção e remoção de elementos da lista

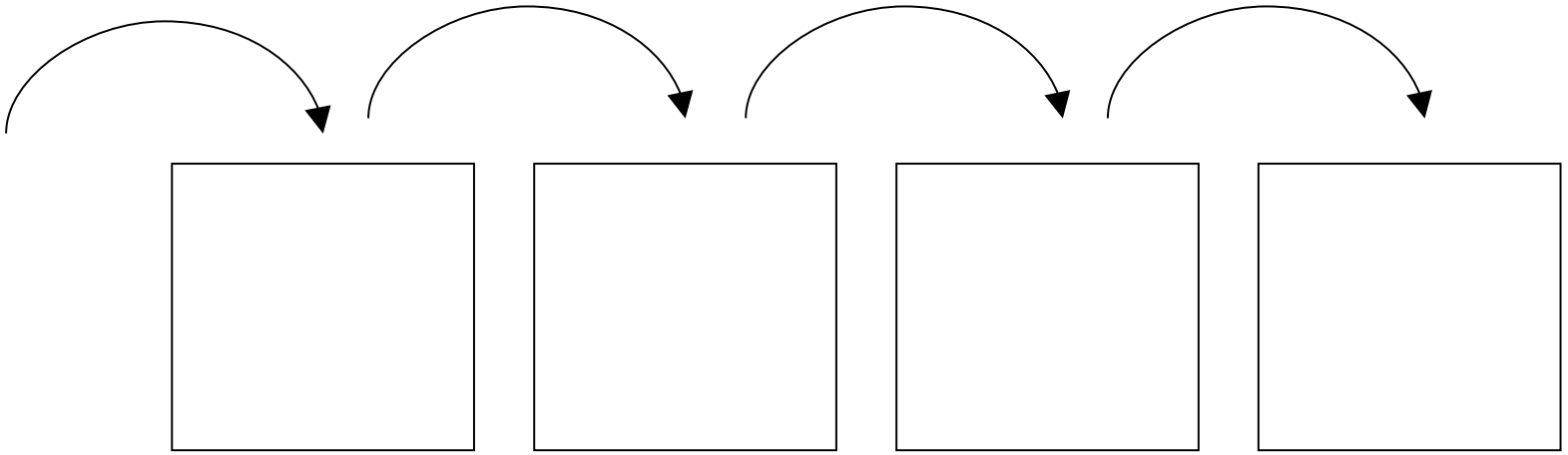
Lista de Objetos (List)

- LinkedList (visão concreta)



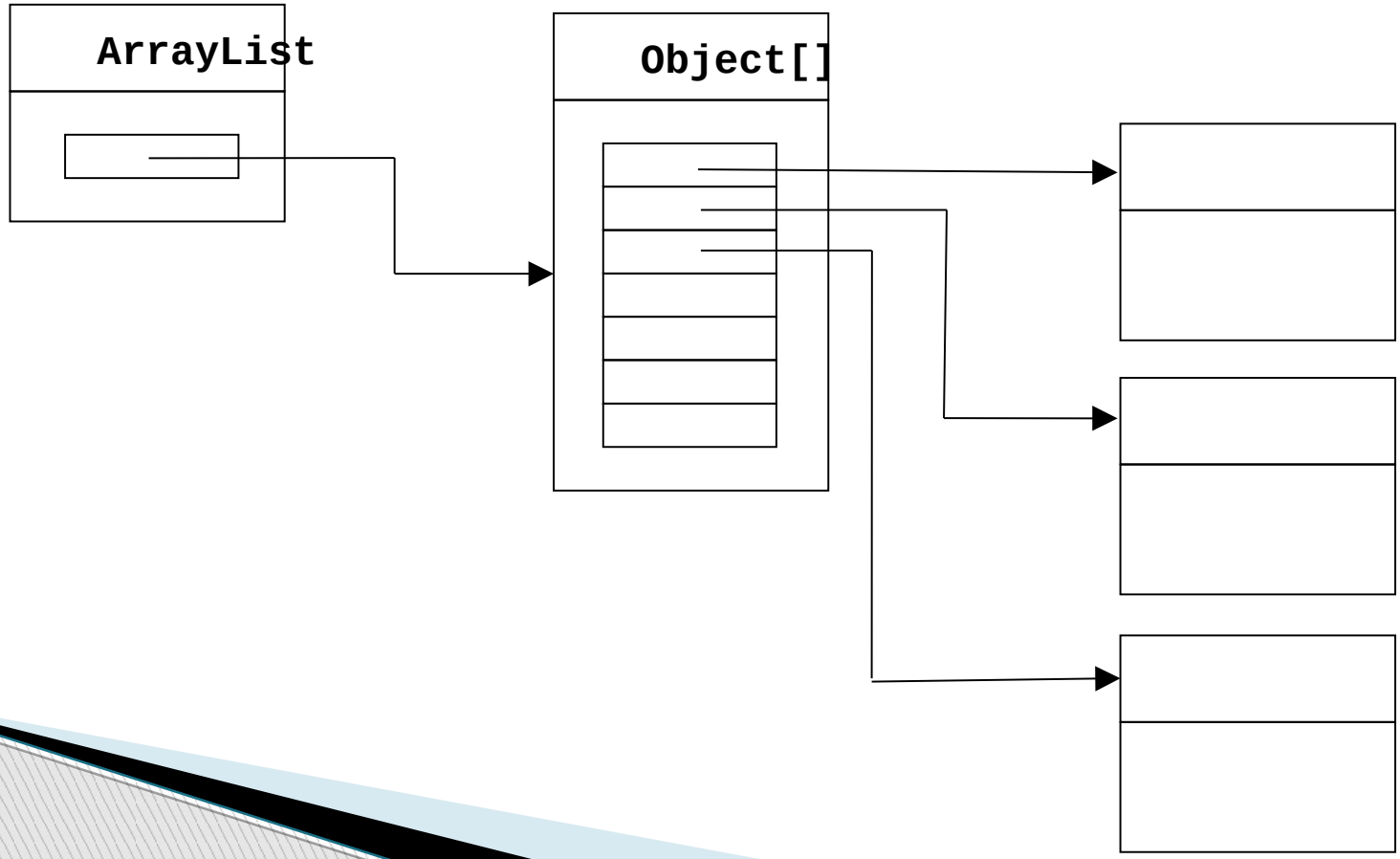
Lista de Objetos (List)

- LinkedList (visão abstrata)
  - Sequência ordenada de itens de dados que podem ser percorridos por um iterador



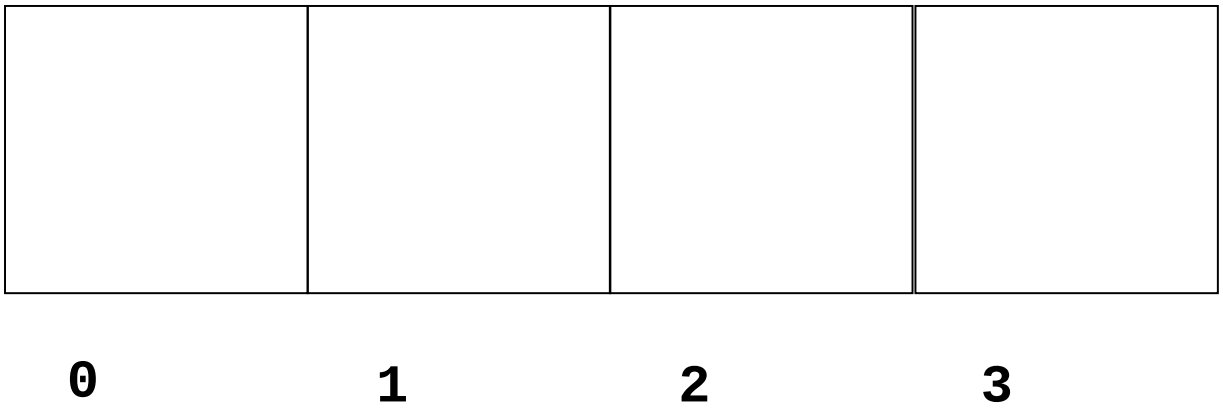
Lista de Objetos (List)

- ArrayList (visão concreta)



Lista de Objetos (List)

- ArrayList (visão abstrata)
  - Sequência ordenada de itens de dados que podem ser acessados com um índice inteiro



Lista de Objetos (LinkedList)

- LinkedList
  - Uso de um iterador de lista para acessar os elementos dentro de uma lista encadeada
  - API Java fornece ListIterator

- ListIterator
  - Encapsula uma posição em qualquer lugar dentro da lista encadeada
  - **listIterator**() – obtém um iterador de lista (aponta para o primeiro elemento)
  - **next**() – mover a posição do iterador
  - **hasNext**() – retorna true se existe um elemento seguinte

Lista de Objetos (LinkedList)

- Exemplo LinkedList e ListIterator (1)

```
import java.util.LinkedList;
import java.util.ListIterator;
public class LinkedListTest {
   public static void main(String[] args) {
       LinkedList<String> pessoas = new LinkedList<>();
       pessoas.addLast("Ronaldo");
       pessoas.addLast("Robinho");
       pessoas.addLast("Rivaldo");
       System.out.println("Impressao da LinkedList:");
       ListIterator<String> iterator = pessoas.listIterator();
       while(iterator.hasNext()){
          String pessoa = iterator.next();
          System.out.println(pessoa);
       }
   }
}
```

Lista de Objetos (ArrayList)

- Exemplo ArrayList (1)

```
import java.util.ArrayList;
import java.util.ListIterator;
public class ArrayListTest {
   public static void main(String[] args) {
       ArrayList<String> pessoas = new ArrayList<>();
       pessoas.add("Ronaldo");
       pessoas.add("Robinho");
       pessoas.add(”Rivaldo");
       System.out.println("Impressao da ArrayList (usando
                          iterador):");
       ListIterator<String> iterator = pessoas.listIterator();
       while(iterator.hasNext()){
           String pessoa = iterator.next();
          System.out.println(pessoa);
       }
   }
}
```

Lista de Objetos (ArrayList)

- Exemplo ArrayList (2)

```
import java.util.ArrayList;
public class ArrayListTest2 {
   public static void main(String[] args) {
       ArrayList<String> pessoas = new ArrayList<>();
       pessoas.add("Ronaldo");
       pessoas.add("Robinho");
       pessoas.add(”Rivaldo");
       System.out.println("Impressao da ArrayList (usando
                          indices):");
       for(int i=0;i<pessoas.size();i++){
           String pessoa = pessoas.get(i);
          System.out.println(pessoa);
       }
   }
}
```

Utilizando for(each)

```
import java.util.ArrayList;
import java.util.ListIterator;

public class ArrayListTest3 {
   public static void main(String[] args) {
      ArrayList<String> pessoas = new ArrayList<>();
      pessoas.add("Ronaldo");
      pessoas.add("Robinho");
      pessoas.add("Rivaldo");
      System.out.println("Impressao da ArrayList (usando for):");

      for(String pessoa : pessoas){
         System.out.println(pessoa);
      }
   }
}
```