

Linguagem de Programação II

IMD0040

Aula 04 – Coleções, TADs e API

Biblioteca de classes Java

- ❑ Várias classes úteis que ajudam na implementação de novos produtos;
- ❑ Não temos que **escrever** tudo do zero;
- ❑ Java chama suas bibliotecas de **pacotes**;
- ❑ O programador Java deve ser capaz de trabalhar com bibliotecas.
- ❑ Agrupar objetos é algo comum:
 - ❖ Pacote **java.util** contém classes que implementam Coleções;
 - ❖ **Java Collections.**

Lendo a documentação

- ❑ Documentação das bibliotecas Java no formato HTML:
 - ❖ Abre em qualquer navegador Web.
- ❑ API: Application Programmers' Interface.
- ❑ Descrição de interface de todas as classes da biblioteca.
- ❑ <https://docs.oracle.com/en/java/javase/index.html>

Lendo a documentação

❑ Acessando a API:

Java Platform, Standard Edition Documentation

Java Platform, Standard Edition (Java SE) helps you develop and deploy Java applications on desktops and servers. Java offers the rich user interface, performance, versatility, portability, and security that today's applications require.

Latest Release

JDK 20

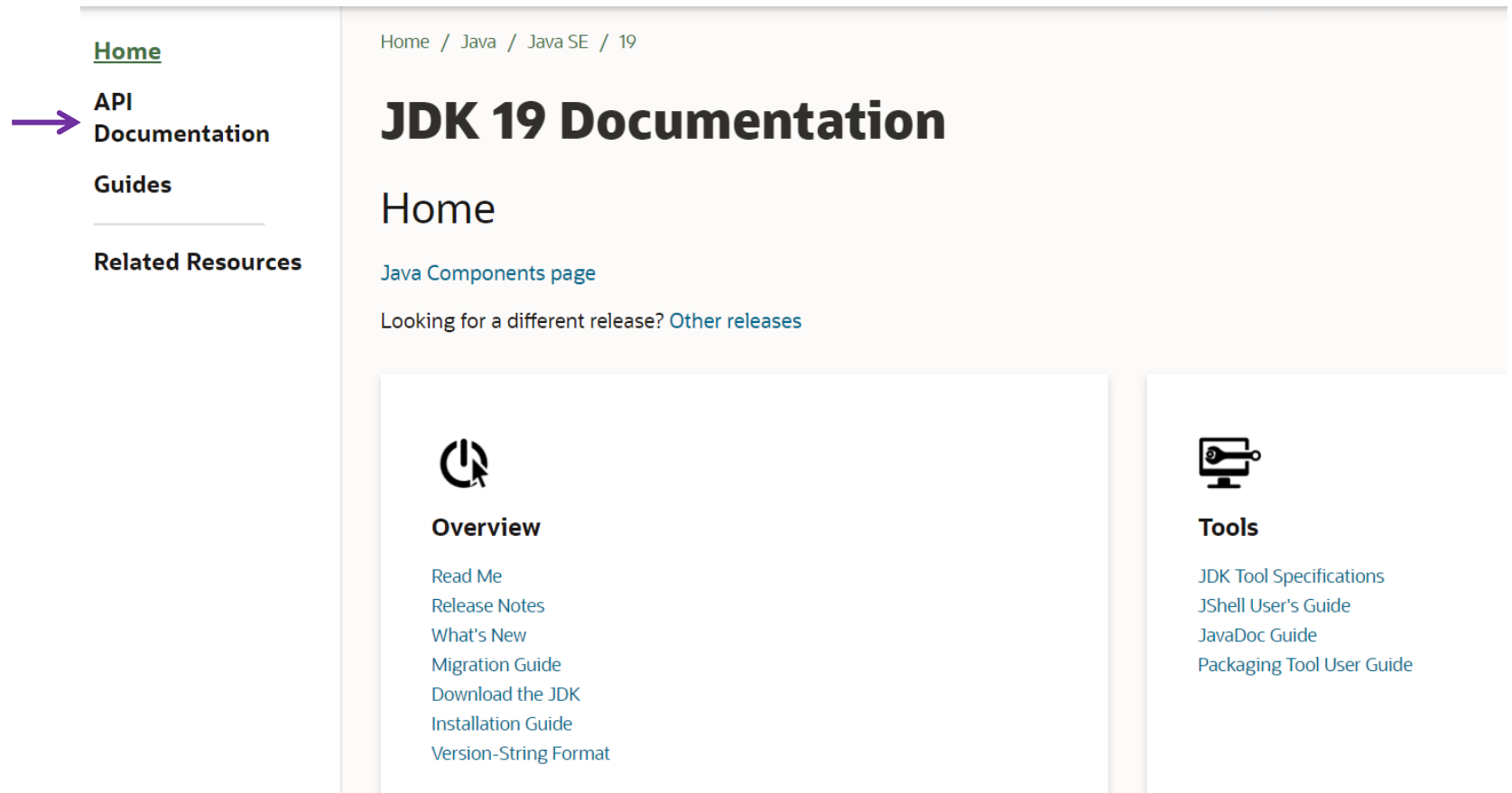
Previous Releases

JDK 19
JDK 18
JDK 17
JDK 16
JDK 15
JDK 14
JDK 13
JDK 12
JDK 11
JDK 10
JDK 9
JDK 8
JDK 7



Lendo a documentação

❑ Acessando a API:



The screenshot displays the official JDK 19 documentation website. On the left, a navigation sidebar contains links for [Home](#), [API Documentation](#) (highlighted with a purple arrow), [Guides](#), and [Related Resources](#). The main content area features a breadcrumb trail: [Home](#) / [Java](#) / [Java SE](#) / 19. Below this, the title **JDK 19 Documentation** is prominently displayed, followed by the word **Home**. A link for [Java Components page](#) is provided, along with a note: "Looking for a different release? [Other releases](#)". The page is divided into two columns. The left column, titled **Overview** with a power button icon, lists links for [Read Me](#), [Release Notes](#), [What's New](#), [Migration Guide](#), [Download the JDK](#), [Installation Guide](#), and [Version-String Format](#). The right column, titled **Tools** with a monitor icon, lists links for [JDK Tool Specifications](#), [JShell User's Guide](#), [JavaDoc Guide](#), and [Packaging Tool User Guide](#).

Lendo a documentação

□ Acessando a API:

[Print](#)

Java Platform, Standard Edition (Java SE) 8

[Home](#) [Client Technologies](#) [Embedded](#) [All Books](#)

About Java SE 8

- [What's New \(Features and Enhancements\)](#)
- [Commercial Features](#) 
- [Compatibility Guide](#)
- [Known Issues](#)

Download and Install

- [Certified System Configurations](#)
- [Download and Installation Instructions](#)


Write Your First Application

- [Get Started with Java](#)
- [Get Started with JavaFX](#)

Learn the Language

- [Java Tutorials Learning Paths](#)

Monitor and Troubleshoot

- [Java Mission Control](#) 
- [Java Flight Recorder](#) 
- [Troubleshooting Guide](#)


HotSpot Virtual Machine

- [HotSpot Virtual Machine Garbage Collection Tuning Guide](#)
- [JRockit to HotSpot Migration Guide](#)

Deploy

- [Deployment Guide](#)

Reference

- [Java SE API Documentation](#) 
- [JavaFX API Documentation](#)
- [Developer Guides](#)
- [Java Language and Virtual Machine Specifications](#)
- [Java SE Tools Reference for UNIX](#)
- [Java SE Tools Reference for Windows](#)

Release Notes

- [Java SE Release Notes](#)

Lendo a documentação

❏ Acessando a API:

The screenshot shows the Oracle Java API documentation website for the `java.util` package. The browser address bar displays `docs.oracle.com/javase/8/docs/api/index.html`. The left sidebar lists various Java packages, with `java.util` highlighted. The main content area shows the `Package java.util` page, which includes a description of the package and a summary of its interfaces.

Overview of the `java.util` package documentation:

- Package:** `java.util`
- Description:** Contains the collections framework, legacy collection classes, event model, tokenizer, a random-number generator, and a bit array.
- See:** Description
- Interface Summary:**
 - `Collection<E>`
 - `Comparator<T>`
 - `Deque<E>`
 - `Enumeration<E>`

Lendo a documentação

□ Acessando a API:

The screenshot shows the Oracle Java API documentation for `java.util.ArrayList`. The browser address bar displays `docs.oracle.com/javase/8/docs/api/index.html`. The left sidebar contains a tree view of the Java API packages, with `java.util` selected and highlighted by a yellow box. Below the package list, the `Classes` section lists various classes, with `ArrayList` highlighted by a yellow box. The main content area shows the documentation for `Class ArrayList<E>`, including its inheritance hierarchy, implemented interfaces, and subclasses. The `compact1, compact2, compact3` methods are also visible at the top of the class page.

← → ↺ docs.oracle.com/javase/8/docs/api/index.html

Apps ★ Bookmarks Portal do Servidor ⓘ OpenRes - Gerenci... Login - Atende IMD Login - Instituto Me... eSocial Editorial Manager®

OVERVIEW PACKAGE **CLASS** USE TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

compact1, compact2, compact3
java.util

Class ArrayList<E>

java.lang.Object
java.util.AbstractCollection<E>
java.util.AbstractList<E>
java.util.ArrayList<E>

All Implemented Interfaces:
Serializable, Cloneable, Iterable<E>, Collection<E>, List<E>, RandomAccess

Direct Known Subclasses:
AttributeList, RoleList, RoleUnresolvedList

```
public class ArrayList<E>  
extends AbstractList<E>  
implements List<E>, RandomAccess, Cloneable, Serializable
```

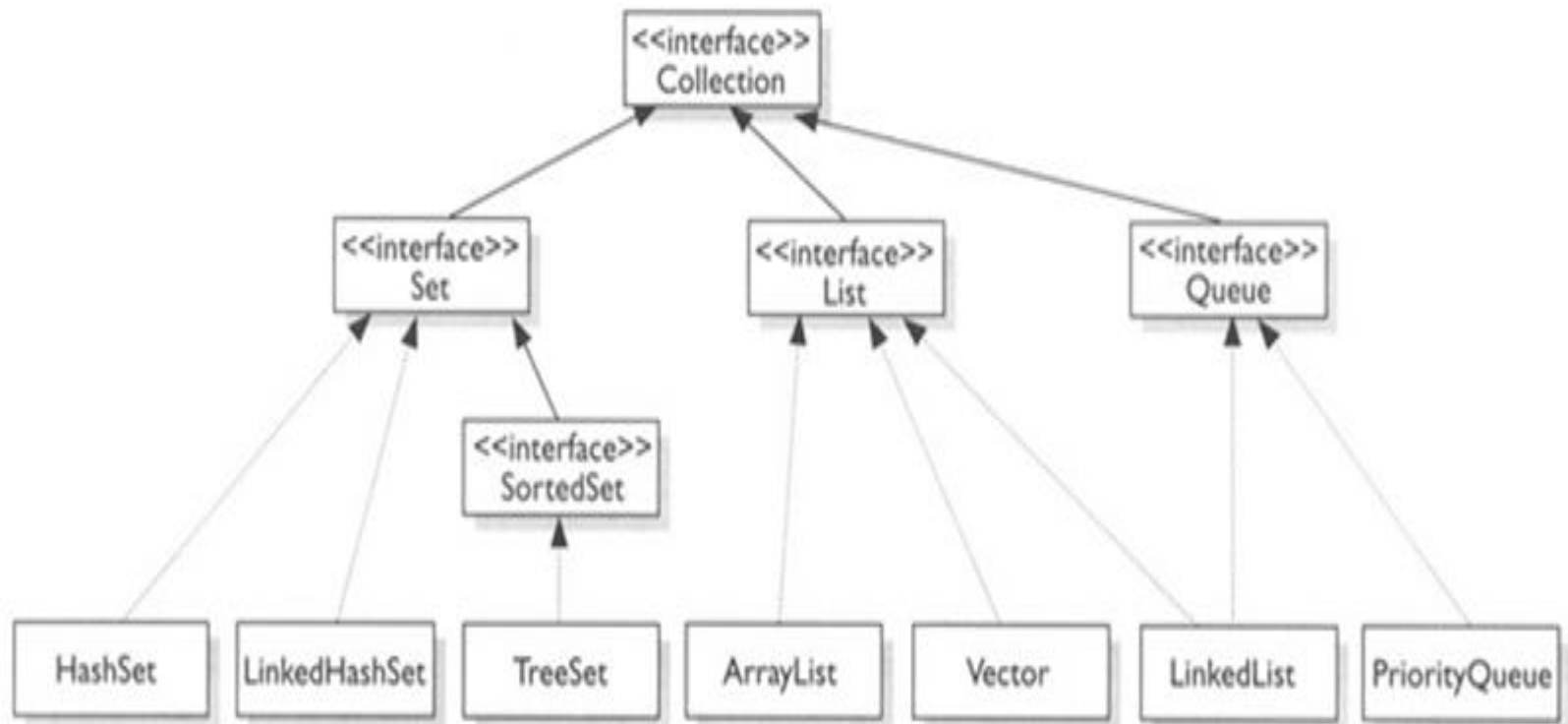

Utilizando classes de biblioteca

- ❑ Classes de biblioteca devem ser importadas:
 - ❖ Uso da palavra **import**.
 - ❖ Exceto classes do pacote **java.lang**

- ❑ Pode se importar uma única classe:
 - ❖ **import java.util.ArrayList**

- ❑ Pode-se importar pacotes inteiros:
 - ❖ **import java.***

Java Collection



<http://www.devmedia.com.br/visao-geral-da-interface-collection-em-java/25822>

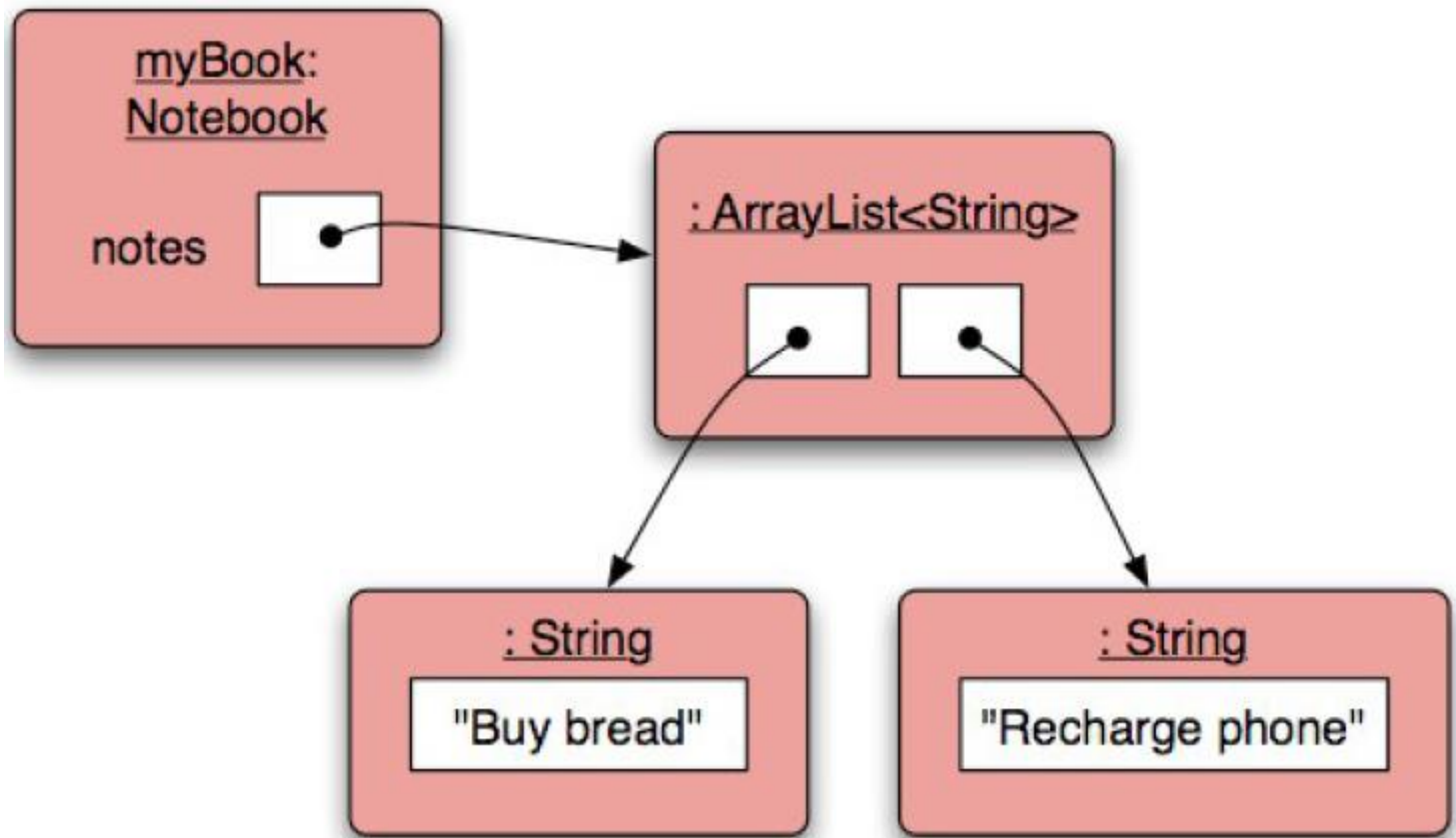
Java Collection

- ❑ É a interface absoluta na hierarquia de coleções. Dela descendem as interfaces **Set**, **Queue** e **List** que formam as coleções genéricas da linguagem Java:
 - ❖ **Set**: define uma coleção que **não contém** valores duplicados.
 - ❖ **Queue**: define uma coleção que representa uma fila, ou seja, implementa o modelo **FIFO** (First-In, First-Out).
 - ❖ **List**: define uma **coleção ordenada** que pode conter **elementos duplicados**.

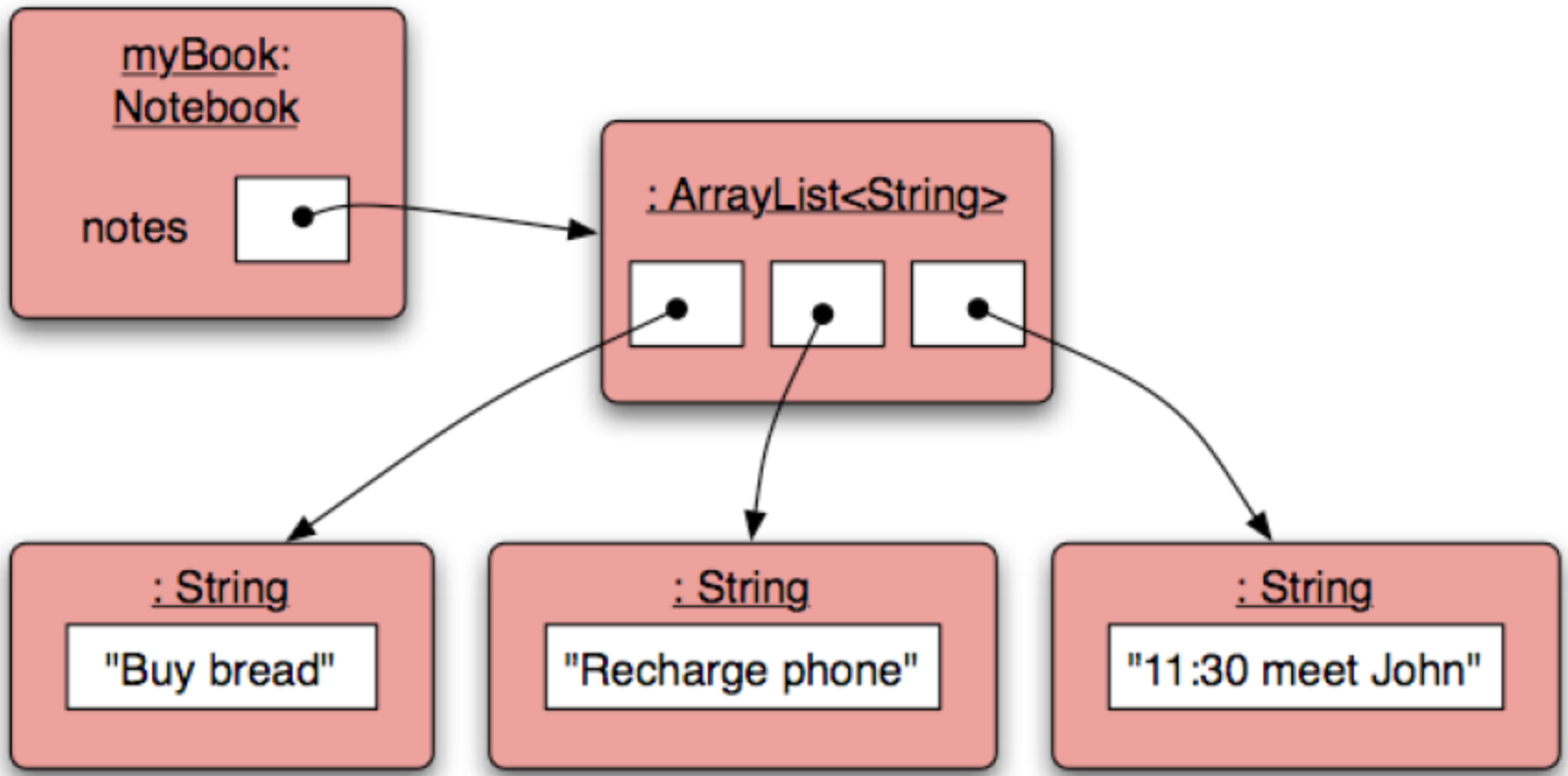
Classe Notebook

```
import java.util.ArrayList;  
  
public class Notebook  
{  
    private ArrayList<String> notes;  
  
    public Notebook()  
    {  
        notes = new ArrayList<String>();  
    }  
}
```

Estruturas de objeto com coleções



Adicionando uma terceira anotação



Recursos de uma coleção

- ❑ Tamanho dinâmico:
 - ❖ Aumenta sua capacidade de acordo com a necessidade.
- ❑ Mantém contagem de items:
 - ❖ Método `size()`.
- ❑ Mantém os objetos em ordem:
 - ❖ Depende do tipo da coleção.
- ❑ Detalhes são ocultos:
 - ❖ Você não precisa saber como funciona para poder usar.

Usando uma coleção

❑ Classe e Métodos:

```
1  import java.util.ArrayList;
2
3  public class Notebook {
4
5      // Storage for an arbitrary number of notes.
6      private ArrayList<String> notes;
7
8      // Metodo construtor
9      public Notebook() {
10         notes = new ArrayList<String>();
11     }
12
13     // Armazena um note
14     public void storeNote(String note) {
15         notes.add(note);
16     }
17
18     // Retorna a quantidade de notes
19     public int numberOfNotes() {
20         return notes.size();
21     }
```

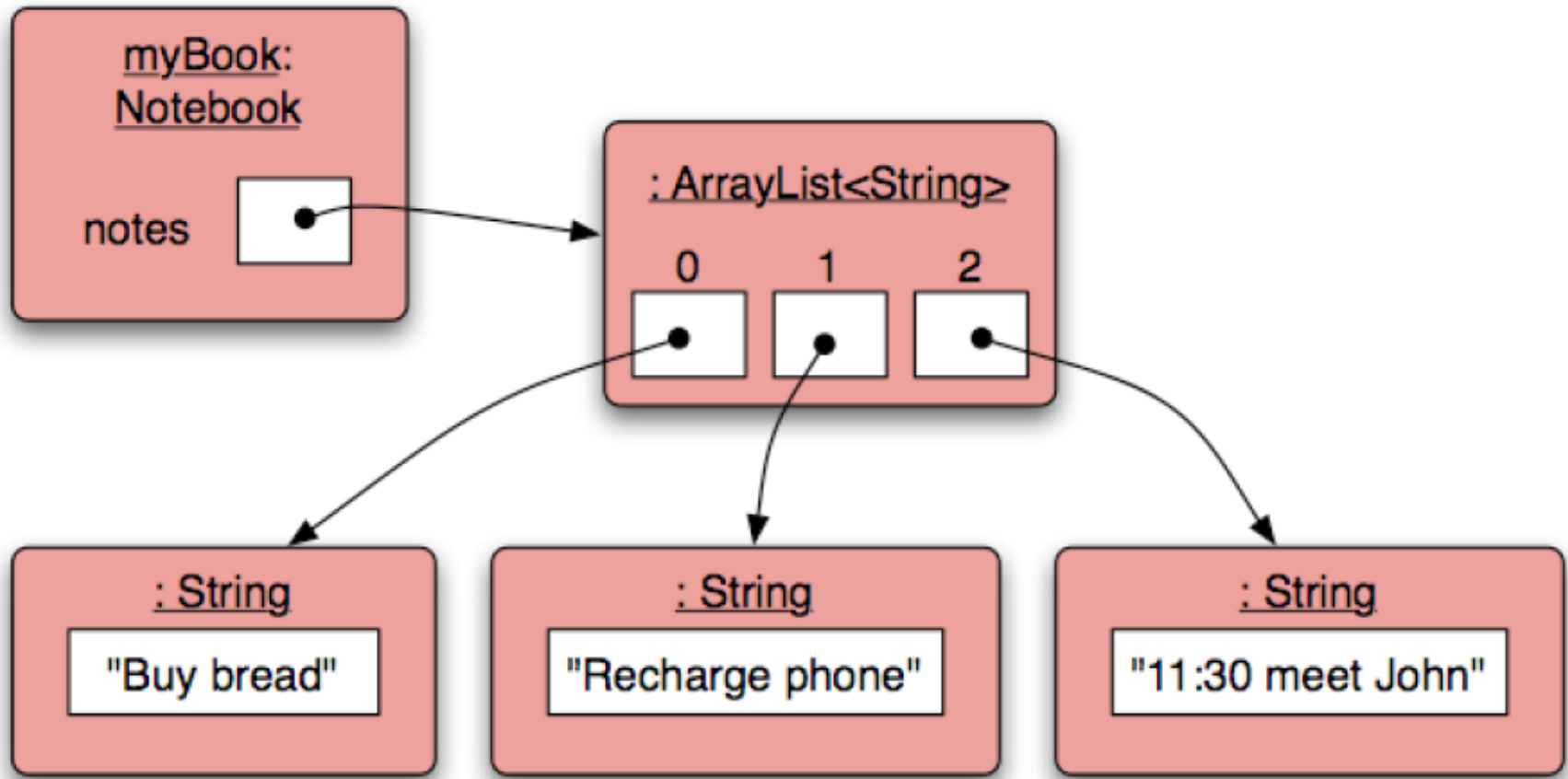

Usando uma coleção

❑ Classe e Métodos:

```
22
23 // Busca por índice
24 public void showNote(int noteNumber) {
25     if(noteNumber < 0) {
26         // This is not a valid note number, so do nothing.
27     }
28     else if(noteNumber < numberOfNotes()) {
29         // This is a valid note number, so we can print it.
30         System.out.println(notes.get(noteNumber));
31     }
32     else {
33         // This is not a valid note number, so do nothing.
34     }
35 }
36 }
```

Usando uma coleção

□ Numeração de índice:



Usando uma coleção

❑ Recuperando um objeto:

```
public void showNote(int noteNumber)
{
    if (noteNumber < 0) {
        //numero invalido
    }
    else if (noteNumber < numberOfNotes()) {
        System.out.println(notes.get(noteNumber));
    }
    else {
        //numero invalido
    }
}
```

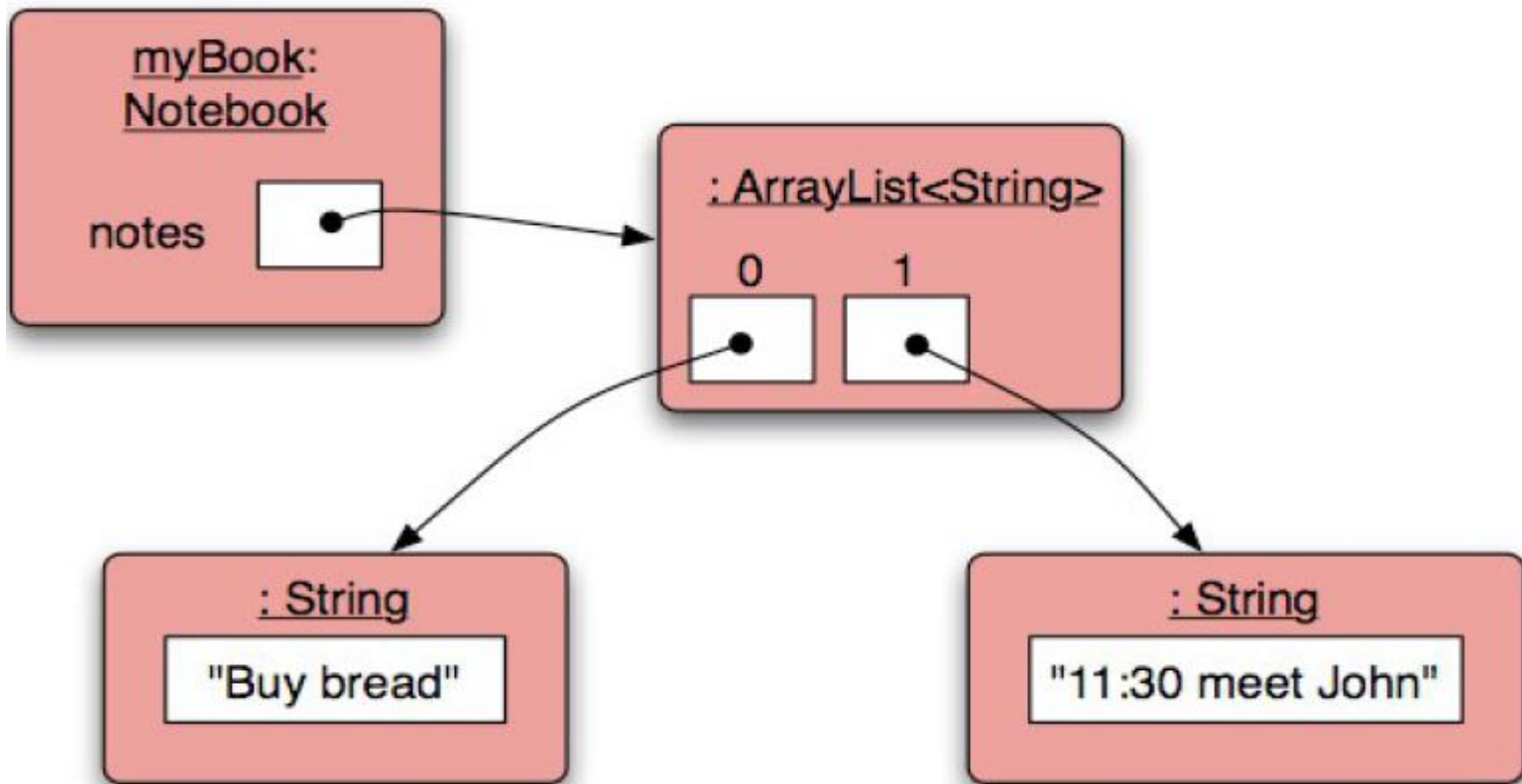
Usando uma coleção

❑ Removendo um objeto:

```
public void removeNote(int noteNumber)
{
    if (noteNumber < 0) {
        //numero invalido
    }
    else if (noteNumber < numberOfNotes()) {
        notes.remove(noteNumber);
    }
    else {
        //numero invalido
    }
}
```

Usando uma coleção

- Remoção afeta a numeração:



Coleções

- ❑ Coleções são conhecidas como tipos **parametrizados** ou **genéricos**:
 - ❖ Funcionam para diversos tipos.
- ❑ **ArrayList** implementa funcionalidades de lista:
 - ❖ add, get, size, etc.
- ❑ Parâmetro de tipo informa o tipo contido na lista:
 - ❖ `ArrayList<Person>`
 - ❖ `ArrayList<TicketMachine>`

Loop

Loop

❑ for-each:

❑ Cabeçalho:

- ❖ Para cada elemento **element** da coleção **collection**, faça ...
- ❖ Atenção para o tipo do elemento.

❑ Corpo:

- ❖ Ações a serem realizadas.

```
for (elementType element : collection) {  
    loop body  
}
```


Loops

□ for-each:

```
/**
 * Lista todas as notas no bloco de notas.
 */
public void listNotes()
{
    for (String note : notes) {
        System.out.println(note);
    }
}
```

Comparação de Strings

- ❑ Comparar duas Strings é a mesma coisa que comparar dois objetos?????

Comparação de Strings

- ❑ Comparar duas Strings é a mesma coisa que comparar dois objetos?
 - ❖ Lembre-se das referências.
- ❑ Quando um objeto é considerado igual ao outro?

Comparação de Strings

- ❑ Comparar duas Strings é a mesma coisa que comparar dois objetos?

- ❖ Lembre-se das referências.

- ❑ Quando um objeto é considerado igual ao outro?

```
if (obj == "bye") {  
    ...  
}  
  
if (obj1 == obj2) {  
    ...  
}
```

Comparação de Strings

- ❑ Deve-se usar o método `equals()`:
 - ❖ `obj.equals(obj1)`.

```
if (obj.equals("bye")) {  
    ...  
}  
  
if (obj1.equals(obj2)) {  
    ...  
}
```

Identidade versus Igualdade

❑ Testando a identidade:

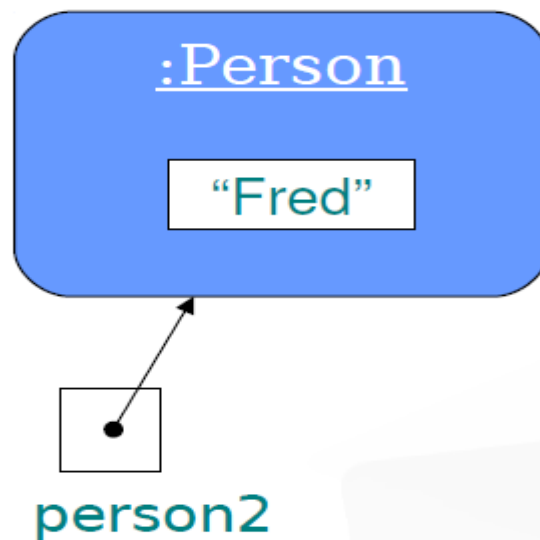
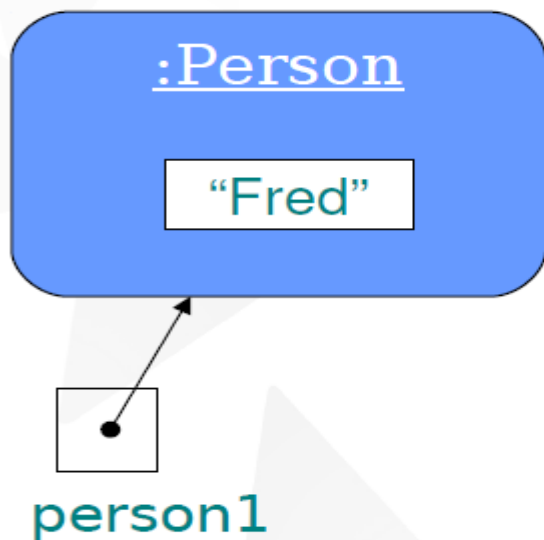
❖ Duas **referências** apontam para um **mesmo objeto**.

❑ Testando a igualdade:

❖ Dois **objetos distintos** (com referências diferentes) possuem o **mesmo estado**.

Identidade versus Igualdade

- ❑ No exemplo abaixo:
 - ❖ **person1 == person2?**



Identidade versus Igualdade

❑ No exemplo abaixo:

❖ **person1 == person2?**

```
public static void main(String args []) {  
  
    Pessoa person1 = new Pessoa();  
    person1.setNome("Fred");  
  
    Pessoa person2 = new Pessoa();  
    person2.setNome("Fred");  
  
    if (person1 == person2) {  
        System.out.print("T");  
    }  
    else {  
        System.out.print("F");  
    }  
}
```

```
PS C:\JCXavier\UFRN\Ano2023-1\IMD0040\Fontes\Lp2_Aula04> java Referencia  
F
```


Identidade versus Igualdade

❑ No exemplo abaixo:

❖ **person1 == person2?**

```
// atribuição de objetos
person2 = person1;

if (person1 == person2) {
    System.out.print("T");
}
else {
    System.out.print("F");
}
```

```
PS C:\JCXavier\UFRN\Ano2023-1\IMD0040\Fontes\Lp2_Aula04> java Referencia
T
```

Identidade versus Igualdade

□ No exemplo abaixo:

❖ **Person1.equals(person2)?**

```
if (person1.getNome().equals(person2.getNome())) {  
    System.out.print("T");  
}  
else {  
    System.out.print("F");  
}
```

```
PS C:\JCXavier\UFRN\Ano2023-1\IMD0040\Fontes\Lp2_Aula04> java Referencia  
T
```

Outras Coleções

Utilizando Conjuntos

```
import java.util.HashSet;  
  
...  
HashSet<String> mySet = new HashSet<String>();  
  
mySet.add("one");  
mySet.add("two");  
mySet.add("three");  
  
for (String element:mySet) {  
    System.out.println(element);  
}
```

- ❑ Um conjunto (set) é uma coleção que armazena cada elemento individual sem repetição. Ele não mantém qualquer ordem específica.

Utilizando Mapas

- ❑ Mapas são coleções que contêm pares de valores:
 - ❖ Chave (K);
 - ❖ Valor (V).
- ❑ Tipos dos pares precisa ser definido pelo usuário.
- ❑ Na inserção precisa informar a chave e o valor.
- ❑ Pesquisa funciona informando a chave e recuperando um valor.
- ❑ Exemplo: lista telefônica.

Utilizando Mapas

❑ Implementação de Lista Telefônica:

```
import java.util.HashMap;

public class ListaTelefonica{

    private HashMap<String, Pessoa> lista;

    public ListaTelefonica() {
        lista = new HashMap<String, Pessoa>();
    }

    public void inserirContato(String numero, Pessoa pessoa) {
        lista.put(numero, pessoa);
        System.out.println("Pessoa inserida na lista!!!!");
    }

    public void buscarContato(String numero) {
        Pessoa pessoa = lista.get(numero);
        if (pessoa != null) {
            System.out.println("Nome: " + pessoa.getNome()
                               + " - email: " + pessoa.getEmail());
        }
        else {
            System.out.println("Numero nao encontrado!!!");
        }
    }
}
```

Utilizando Mapas

❑ Implementação de Lista Telefônica:

```
public class ListaTelefonicaView{  
    public static void main(String args []){  
  
        Pessoa pessoa1 = new Pessoa();  
        pessoa1.setNome("Roberto Carlos");  
        pessoa1.setEmail("roberto@yahoo.com");  
        String numero1 = "(084)99148-7898";  
  
        Pessoa pessoa2 = new Pessoa();  
        pessoa2.setNome("Maria dos Anjos");  
        pessoa2.setEmail("maria@yahoo.com");  
        String numero2 = "(084)98155-1015";  
  
        // cadastrar pessoas na lista  
        ListaTelefonica lista = new ListaTelefonica();  
        lista.inserirContato(numero1, pessoa1);  
        lista.inserirContato(numero2, pessoa2);  
  
        // buscar pessoa na lista  
        String busca = "(084)99148-7898";  
        lista.buscarContato(busca);  
  
        System.out.println();  
        busca = "(084)99948-0008";  
        lista.buscarContato(busca);  
    }  
}
```

Utilizando Mapas

❑ Implementação de Lista Telefônica:

```
PS C:\JCXavier\UFRN\Ano2022-2\IMD0040\Fontes\Lp2_Aula04> java ListaTelefonicaView
Pessoa inserida na lista!!!
Pessoa inserida na lista!!!
Nome: Roberto Carlos - email: roberto@yahoo.com

Numero nao encontrado!!!
```


Perguntas ...



Obrigado!!!

