

Considerações sobre o uso de String em Java

Disciplina: Linguagem de Programação



Introdução

- Uma string é uma sequência de caracteres
- Em Java uma string é um objeto, uma instância da classe String do pacote `java.lang`
- Strings em Java são constantes (imutáveis), ou seja, uma vez criadas elas não podem ser alteradas
- As constantes do tipo `String` aparecem entre aspas

Introdução

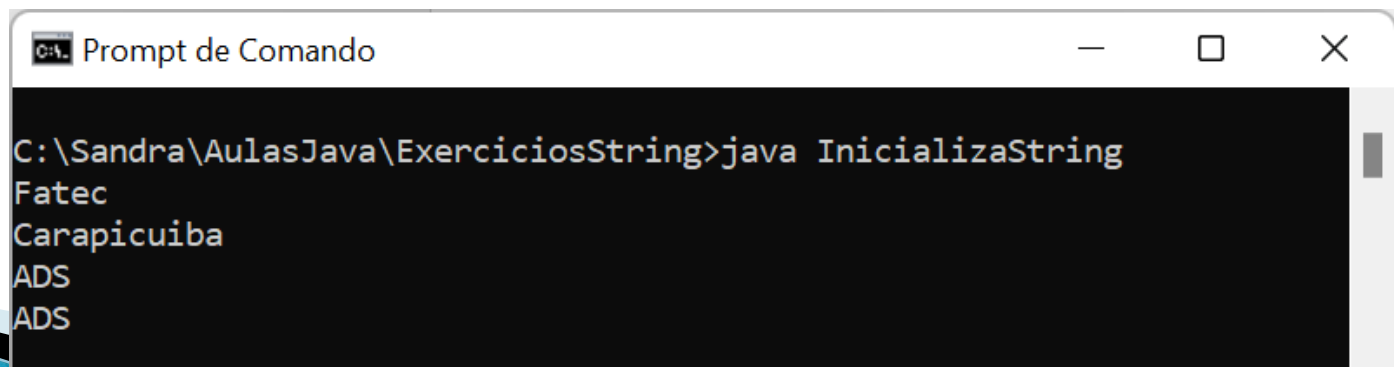
- Imutáveis?
- Segundo artigo da Devmedia
 - Strings são imutáveis, ou seja, não é possível mudar o seu valor após a primeira atribuição.
 - Mas se as strings são imutáveis, como conseguimos concatenar vários valores a uma string?
 - O que acontece é que não estamos concatenando nada e sim criando novos objetos na memória.
 - As strings antigas perdem a referência mas continuam lá.

Criação de uma String

- Para declarar uma string, usamos o tipo String
- Exemplo:
 - String nome;
- Para atribuir uma String à variável podemos usar um dos construtores da classe String ou uma constante ou variável do tipo String

Exemplo da criação de Strings

```
1 public class InicializaString {  
2  
3     public static void main(String[] args) {  
4  
5         String str1 = new String("Fatec");  
6         String str2 = "Carapicuiba";  
7         char[] caracteres = { 'A', 'D', 'S' };  
8         String str3 = new String(caracteres);  
9  
10        byte[] codigos = { 65, 68, 83 };  
11        String str4 = new String(codigos);  
12  
13        System.out.println(str1);  
14        System.out.println(str2);  
15        System.out.println(str3);  
16        System.out.println(str4);  
17    }  
18 }  
19  
20
```



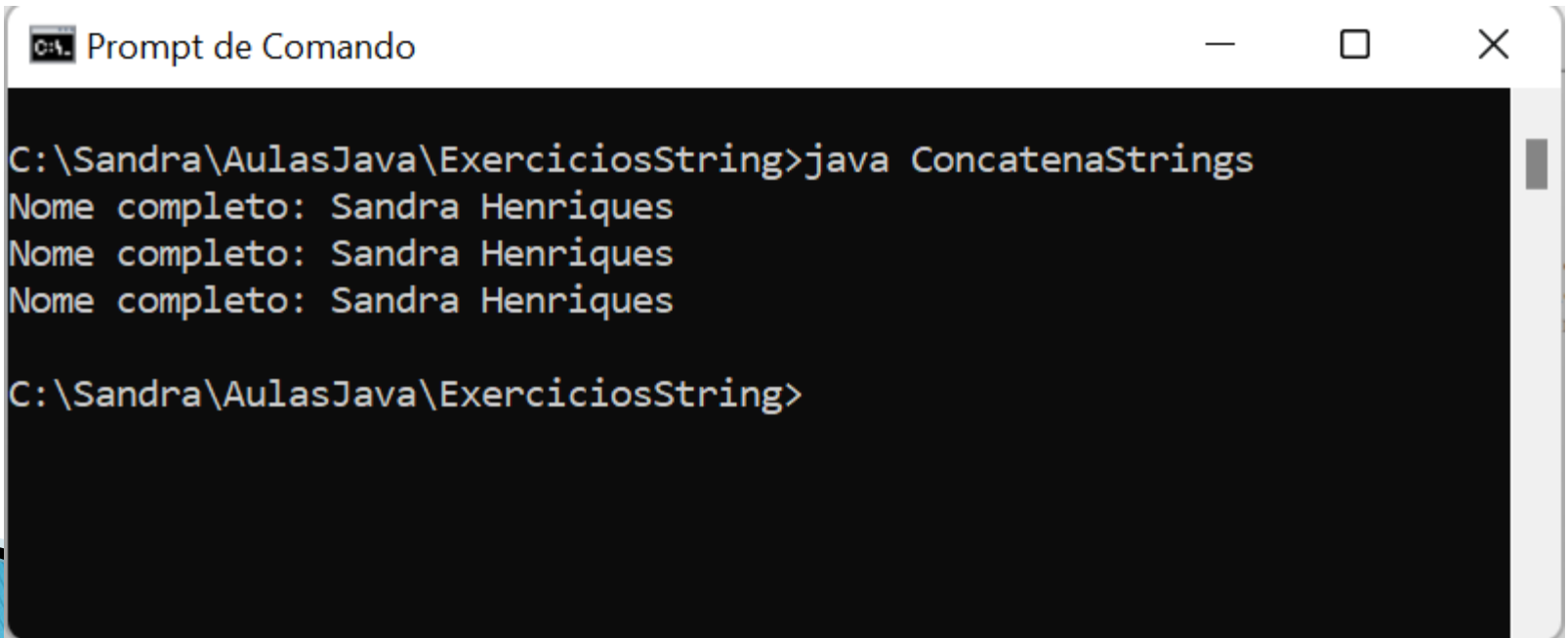
The screenshot shows a Windows Command Prompt window titled "Prompt de Comando". The command entered is `C:\Sandra\AulasJava\ExerciciosString>java InicializaString`. The output of the program is displayed on the following lines: `Fatec`, `Carapicuiba`, `ADS`, and `ADS`.

Concatenação de Strings

- Strings podem ser concatenadas usando o operador +
- Também é possível usar métodos como `concat()` e `format()`

Exemplo de concatenação de Strings

```
1 public class ConcatenaStrings {  
2  
3     public static void main(String[] args) {  
4  
5         String nome = "Sandra";  
6         String sobrenome = "Henriques";  
7         System.out.println("Nome completo: " + nome + " " + sobrenome);  
8         System.out.println("Nome completo: ".concat(nome).concat(" ").concat(sobrenome));  
9         System.out.println(String.format("%s %s %s", "Nome completo:", nome, sobrenome));  
10    }  
11 }  
12
```



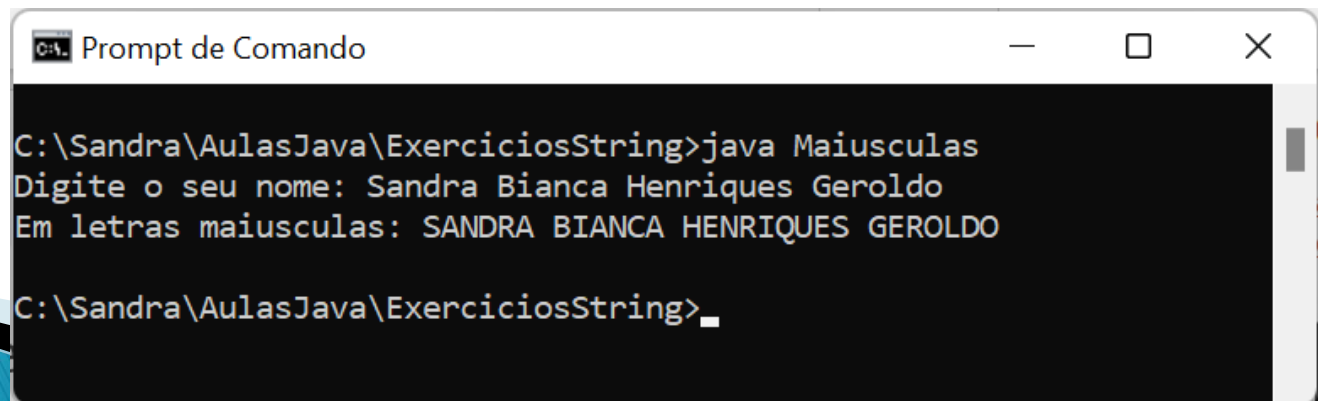
The screenshot shows a Windows Command Prompt window titled "Prompt de Comando". The command executed is `java ConcatenaStrings`. The output of the program is displayed on three separate lines: `Nome completo: Sandra Henriques`, `Nome completo: Sandra Henriques`, and `Nome completo: Sandra Henriques`. The prompt then returns to `C:\Sandra\AulasJava\ExerciciosString>`.

```
C:\Sandra\AulasJava\ExerciciosString>java ConcatenaStrings  
Nome completo: Sandra Henriques  
Nome completo: Sandra Henriques  
Nome completo: Sandra Henriques  
  
C:\Sandra\AulasJava\ExerciciosString>
```

Método toUpperCase()

- ▶ Um string pode ser convertida para letras maiúsculas usando o método toUpperCase()
- ▶ Sintaxe: nomeDoObjeto.toUpperCase();

```
1  import java.util.Scanner;
2  public class Maiusculas {
3
4      public static void main(String[] args) {
5
6          Scanner entra = new Scanner(System.in);
7          System.out.print("Digite o seu nome: ");
8          String nome = entra.nextLine();
9          String maiusculas = nome.toUpperCase();
10         System.out.println("Em letras maiusculas: " + maiusculas);
11     }
12 }
13
14
15
```



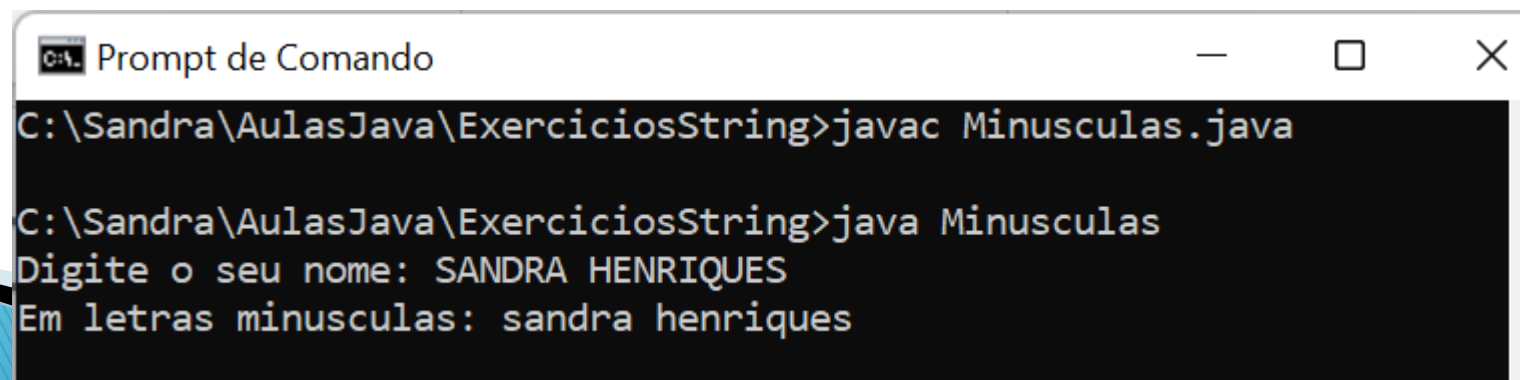
The screenshot shows a Windows Command Prompt window titled "C:\ Prompt de Comando". The command prompt displays the following text:

```
C:\Sandra\AulasJava\ExerciciosString>java Maiusculas
Digite o seu nome: Sandra Bianca Henriques Geroldo
Em letras maiusculas: SANDRA BIANCA HENRIQUES GEROLDO
C:\Sandra\AulasJava\ExerciciosString>_
```


Método toLowerCase()

- ▶ Um string pode ser convertida para letras minúsculas usando o método toLowerCase()
- ▶ Sintaxe : nomeDoObjeto.toLowerCase();

```
1  import java.util.Scanner;
2  public class Minusculas {
3
4      public static void main(String[] args) {
5
6          Scanner entra = new Scanner(System.in);
7          System.out.print("Digite o seu nome: ");
8          String nome = entra.nextLine();
9          String minusculas = nome.toLowerCase();
10         System.out.println("Em letras minusculas: " + minusculas);
11     }
12 }
13
14
```



The screenshot shows a Windows Command Prompt window titled "Prompt de Comando". The command prompt is open at the directory "C:\Sandra\AulasJava\ExerciciosString". The user has entered the command "javac Minusculas.java" to compile the Java file. The output shows the compilation was successful. Then, the user entered the command "java Minusculas" to run the program. The program prompts the user to "Digite o seu nome:" and the user has entered "SANDRA HENRIQUES". The program then outputs "Em letras minusculas: sandra henriques", demonstrating the effect of the toLowerCase() method.

```
C:\Sandra\AulasJava\ExerciciosString>javac Minusculas.java

C:\Sandra\AulasJava\ExerciciosString>java Minusculas
Digite o seu nome: SANDRA HENRIQUES
Em letras minusculas: sandra henriques
```

Comparação de Strings

- Não é recomendado usar o operador `==` para comparar Strings (ou qualquer outro objeto)
- Preferencialmente use os métodos
 - `equals()` e `equalsIgnoreCase()`: devolvem verdadeiro ou falso
 - `compareTo()` e `compareToIgnoreCase()`: devolvem um inteiro

`str1.compareTo(str2)`

- `str1` igual a `str2`: 0
- `str1` menor que `str2`: < 0
- `str1` maior que `str2`: > 0

Exemplo de comparação de Strings

```
1 public class ComparaStrings {  
2  
3     public static void main(String[] args) {  
4  
5         String str1 = new String("Fatec"), str2 = "Fatec", str3 = "fatec";  
6         System.out.println("str1: " + str1 + '\n' + "str2: " + str2 + '\n' + "str3: " + str3);  
7         System.out.println("str1 == str2: " + (str1 == str2));  
8         System.out.println("str1.equals(str2): " + str1.equals(str2));  
9         System.out.println("str1.equals(str3): " + str1.equals(str3));  
10        System.out.println("str1.equalsIgnoreCase(str3): " + str1.equalsIgnoreCase(str3));  
11        System.out.println("str1.compareTo(str2): " + str1.compareTo(str2));  
12        System.out.println("str1.compareTo(str3): " + str1.compareTo(str3));  
13        System.out.println("str3.compareTo(str2): " + str3.compareTo(str2));  
14        System.out.println("str3.compareToIgnoreCase(str2): " + str3.compareToIgnoreCase(str2));  
15    }  
16 }
```

C:\> Prompt de Comando

C:\Sandra\AulasJava\ExerciciosString>javac ComparaStrings.java

C:\Sandra\AulasJava\ExerciciosString>java ComparaStrings

str1: Fatec

str2: Fatec

str3: fatec

str1 == str2: false

str1.equals(str2): true

str1.equals(str3): false

str1.equalsIgnoreCase(str3): true

str1.compareTo(str2): 0

str1.compareTo(str3): -32

str3.compareTo(str2): 32

str3.compareToIgnoreCase(str2): 0

Descobrimos o número de caracteres de uma String

- Para obter o número de caracteres de uma String podemos usar o método `length()`
- Exemplo:
- `String str = "Fatec Carapicuíba";`
 - `System.out.println("Comprimento: "+ str.length());`

Comprimento: 17

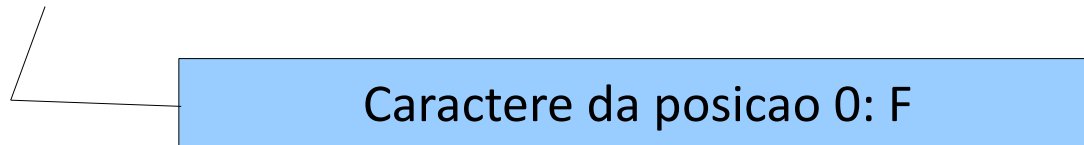
```
1 public class ContaTamanho {  
2  
3     public static void main(String[] args) {  
4  
5         String str = new String("Fatec Carapicuíba");  
6         System.out.println("Comprimento: "+ str.length());  
7  
8     }  
9  
10 }
```

```
C:\Sandra\AulasJava\ExerciciosString>javac ContaTamanho.java
```

```
C:\Sandra\AulasJava\ExerciciosString>java ContaTamanho  
Comprimento: 17
```

Obtendo o caractere de uma posição na String

- Para recuperar o caractere de uma posição específica de uma String, usamos o método `charAt()`
- Exemplo:
- `String str = "Fatec Carapicuíba";`
- `System.out.println("Caractere da primeira posicao ou indice 0: " + str.charAt(0));`



Caractere da posicao 0: F

- Cabe lembrar que a primeira letra é encontrada da posição ou índice 0.

Exercícios

- 1) Escreva um programa que leia duas strings (dois nomes) e as imprima na tela. Imprima também a segunda letra de cada string.
- 2) Crie um programa para ler o primeiro nome de uma pessoa e contar quantas vogais esse nome possui.
- 3) Escreva um programa que leia uma string e substitua todos os caracteres 'a' da string lida por '*'. Informe na tela quantos caracteres foram retirados.

<https://docs.oracle.com/en/java/javase/13/docs/api/index.html>

- ▶ Após acessar o link:

<https://docs.oracle.com/en/java/javase/13/docs/api/index.html>

- ▶ Selecionar as opções:

- [java.base](#)

- [java.lang](#)

- [String](#):

<https://docs.oracle.com/en/java/javase/13/docs/api/java.base/java/lang/String.html>

- Ou

- [StringBuffer](#):

<https://docs.oracle.com/en/java/javase/13/docs/api/java.base/java/lang/StringBuffer.html>

Referências

- DEITEL, H.M.; DEITEL, P.J.. *Java Como Programar*. 4. ed., Porto Alegre: Bookman, 2002.
- ECKEL, B.. *Thinking in Java*. 3. ed., Prentice Hall, 2002.