



## Unidade 4

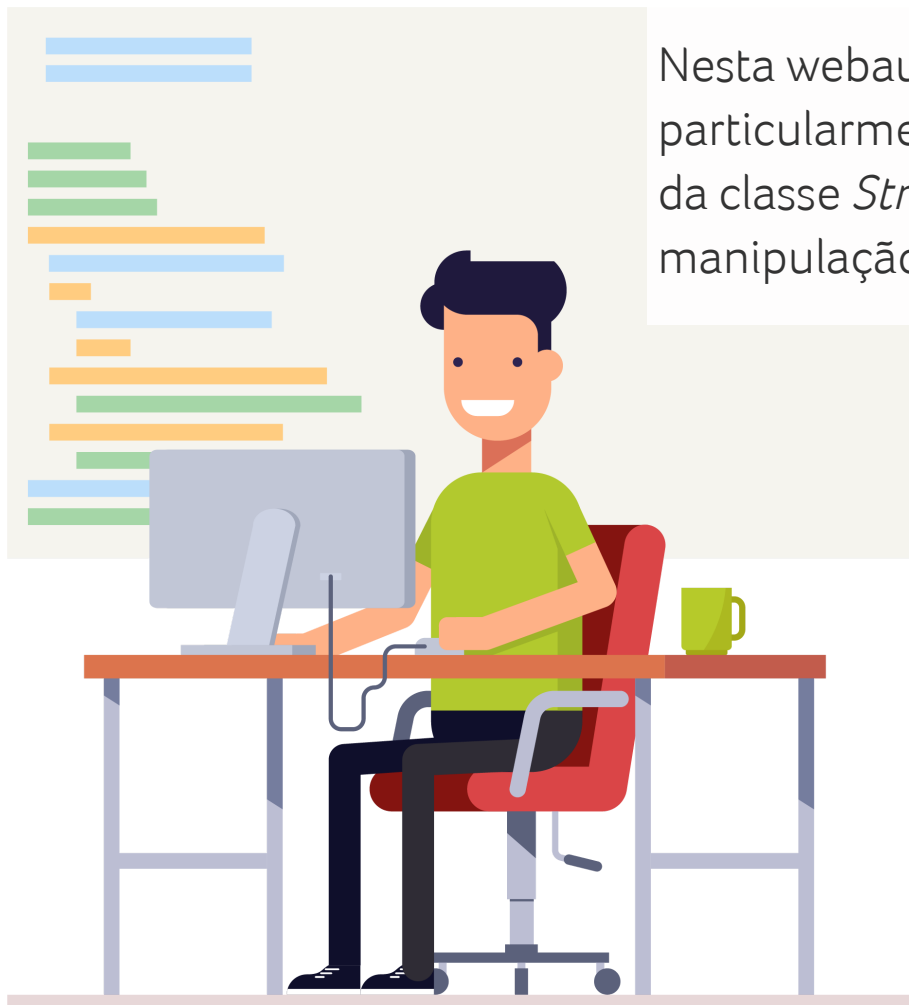
### Seção 2



# Programação Orientada a Objetos

# Webaula 2

## *Strings* em Java



Nesta webaula, serão abordadas as cadeias de caracteres, particularmente chamadas de strings. Por meio dos métodos da classe *String*, é possível o armazenamento e a manipulação de cadeias de caracteres.

## *String*

Uma *string* é uma sequência de caracteres tratada com uma única unidade, que pode incluir letras, dígitos e vários caracteres especiais, como +, -, \*, / e \$. Em Java, as *strings* são tratadas como objetos da classe `java.lang.String` e, por isso, devem ser declaradas e depois instanciadas.

*String* é um tipo texto que corresponde à união de um conjunto de caracteres. Em Java, uma variável do tipo *string* é uma instância da classe `String`, isso é, gera objetos que possuem propriedades e métodos, diferente dos tipos primitivos como `int`, `float`, `double`, etc.

Para efeito de declaração de uma *string*, a sequência de caracteres deve ser atribuída a uma referência da classe `String`. Por exemplo, a linha `String nomeDisciplina="Java"` inicializa `nomeDisciplina`, que servirá para referenciar um objeto da classe `String` que contém a sequência de caracteres "Java".

[Clique aqui para saber mais](#)

Da mesma forma que os *arrays*, cada elemento de uma *string* também é referenciado por um índice, que indica a posição de determinado caractere na sequência de caracteres. Este índice pode valer entre zero e o valor do comprimento da *string* menos um.

- Um objeto da classe `String` não poderá ser alterado depois de criado, já que o Java não oferece nenhuma funcionalidade que mude uma sequência de caracteres.
- Não há necessidade de especificar o tamanho (ou dimensão) de um objeto `String` no ato da sua criação.

## Apresentação da classe *String*

```
public class ConstrutoresString {  
    public static void main (String[] args) {  
        char[] ArrayDeChar = {'a','n','i','v','e','r',  
        's','á','r','i','o'};  
        String s = new String("Olá! Feliz");  
        String s1 = new String();  
        String s2 = new String(s);  
        String s3 = new String(ArrayDeChar);  
        System.out.printf(" s1 = %s\n s2 = %s\n s3 = %s\n",  
s1, s2, s3);  
    }  
}
```

Fonte: autor

Embora a criação de instâncias da classe *String* dispense a chamada do construtor, os objetos ainda assim podem ser criados pela forma usual, por meio do uso da palavra reservada *new* em conjunto com um construtor que recebe como argumento outra instância previamente criada da classe *String*.

Explore a galeria para saber mais.

Os métodos da classe *String* oferecem ao desenvolvedor meios de processar os caracteres que compõem a *string*, usados na comparação entre sequências, na concatenação de duas *strings* e em testes dos mais variados. Explore a galeria para conhecer alguns destes.

length ()

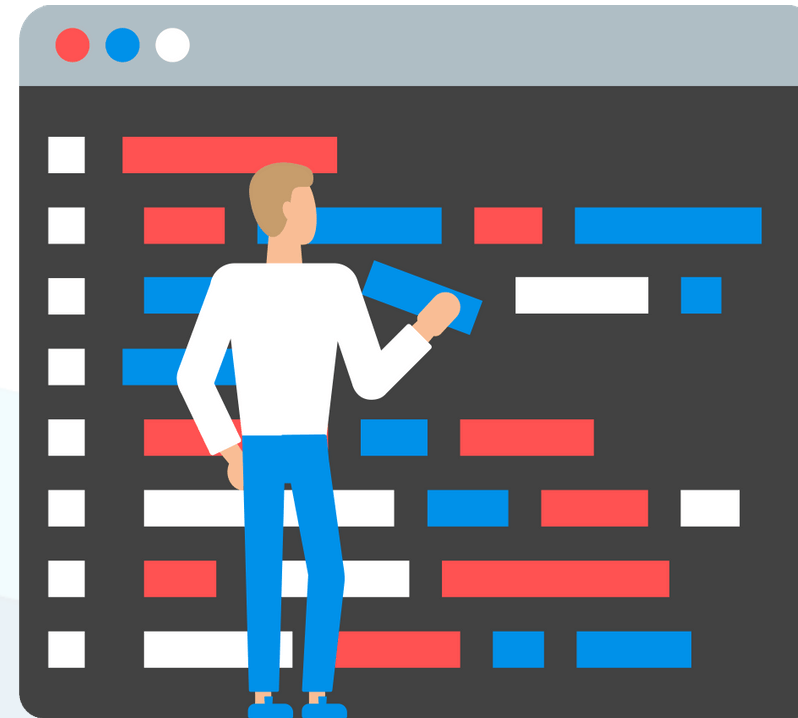
Retorna o tamanho da *string* expresso em um número inteiro.

```
import java.lang.String;
import java.util.Scanner;
public class ValidaEntrada {
    public static void main (String[] args) {
        Scanner entrada = new Scanner(System.in);
        String cpf = new String();
        System.out.print("Informe o número do seu CPF:");
        cpf = entrada.nextLine();
        if (cpf.length() != 11) System.out.println("CPF
        inválido");
    }
}
```

Fonte: autor

## Classe StringBuffer

O Java ainda disponibiliza a classe `StringBuffer`, que permite a criação e manipulação de uma *string* modificável. Uma de suas características mais úteis é a possibilidade de se criar instâncias com um número inicial fixo de caracteres. Esta quantidade inicial pode ser modificada pelo programador ou automaticamente, dependendo da necessidade de crescimento da *string*.

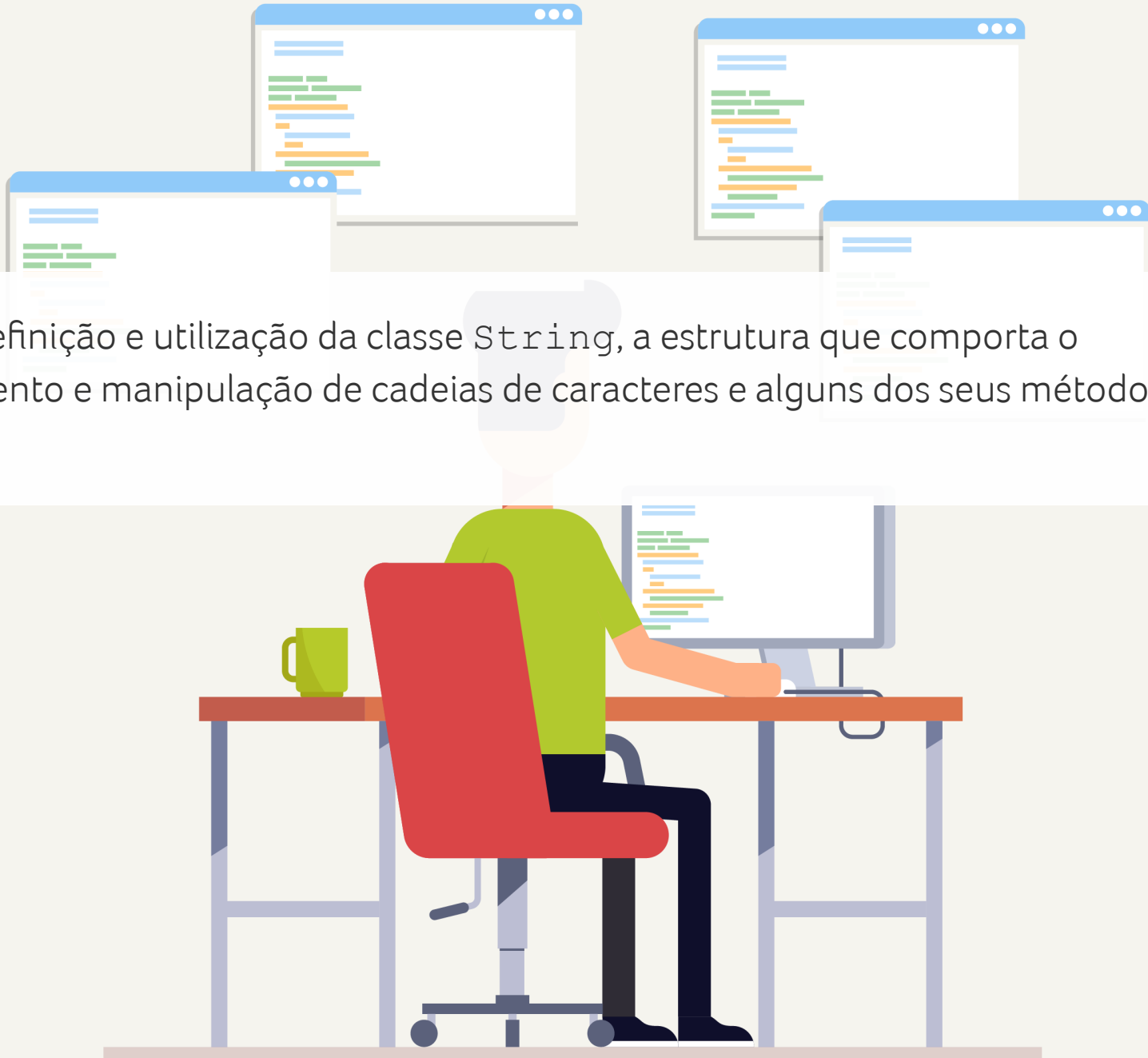




As operações principais dessa classe estão disponíveis nos métodos que realizam inserção e concatenação entre sequências de caracteres. Os métodos de inserção adicionam os caracteres em um ponto específico da *string* e a concatenação se dá no final da sequência.

```
import java.lang.String;
public class StringBufferExemplo {
    public static void main (String[] args) {
        StringBuffer nome = new StringBuffer("José - ");
        nome.append("Gerente ");
        nome.append("Sênior");
        System.out.printf("O nome e o cargo do
funcionário são: %s", nome);
    }
}
```

Fonte: autor

An illustration of a person with grey hair, wearing a green shirt and dark pants, sitting in a red office chair at a wooden desk. They are looking at a computer monitor displaying code. On the desk is a green mug. In the background, there are four floating windows, each showing snippets of code with green, blue, and orange syntax highlighting. The entire scene is set against a light beige background.

Você viu a definição e utilização da classe `String`, a estrutura que comporta o armazenamento e manipulação de cadeias de caracteres e alguns dos seus métodos mais utilizados.

## Você já conhece o Saber?

Aqui você tem na palma da sua mão a **biblioteca digital** para sua **formação profissional**.

Estude no celular, tablet ou PC em qualquer hora e lugar sem pagar mais nada por isso.

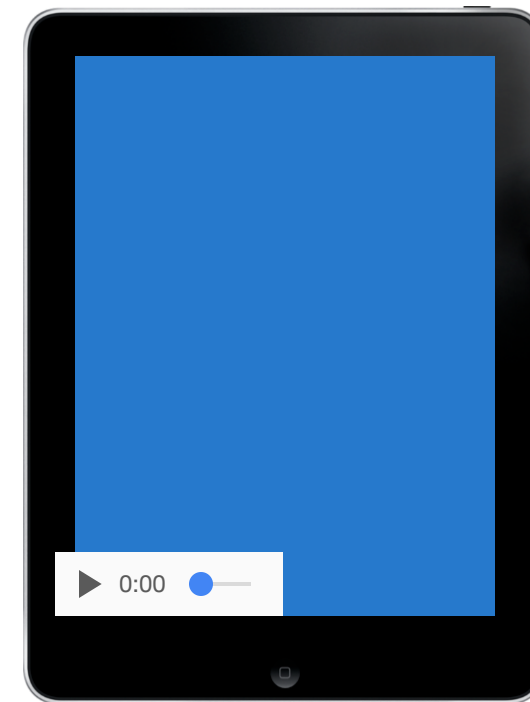
Mais de 450 livros com interatividade, vídeos, animações e jogos para você.



Android:  
<https://goo.gl/yAL2Mv>



iPhone e iPad - IOS:  
<https://goo.gl/OFWqcq>





Bons estudos!

