STRING MÉTODOS ÚTEIS

PRINCIPAIS MÉTODOS PARA O DIA A DIA

```
String nome = "Daniel";
String nome2 = "Daniel";
```

COMPARAÇÃO COM ==

APENAS VERIFICA SE AS DUAS VARIÁVEIS ESTÃO NO MESMO LUGAR DA MEMÓRIA

```
System.out.println(nome == nome2);
```

COMPARAÇÃO COM EQUALS

COMPARA O CONTEÚDO DOS OBJETOS COM BASE NO EQUALS

```
System.out.println(nome.equals(nome2));
```

CONCATENAÇÃO COM CONCAT

VERSÃO 1 PARA CONCATENAÇÃO

```
nome = nome.concat(" Abella");
System.out.println(nome);
```

CONCATENAÇÃO COM OPERADOR +

VERSÃO 2 PARA CONCATENAÇÃO

```
nome = nome + " Abella";
System.out.println(nome);
```

TAMANHO COM LENGTH

CONTA TODOS OS CARACTERES DA STRING, INCLUINDO ESPAÇOS

```
System.out.println(nome.length());
```

CONVERTER PARA MAIÚSCULO E MINÚSCULO

USANDO OS MÉTODOS TOUPPERCASE E TOLOWERCASE

```
nome = nome.toUpperCase();
System.out.println(nome);

nome = nome.toLowerCase();
System.out.println(nome);
```

CONTÉM UMA DADA STRING DENTRO DE OUTRA

POR EXEMPLO, VERIFICAR SE UM DADO NOME POSSUI “ABELLA” COMO SOBRENOM

```
boolean ehDaFamiliaAbella = nome.contains("Abella");
System.out.println(ehDaFamiliaAbella);
```

SUBSTITUIR CARACTERE(S) POR OUTRO(S)

ESTE MÉTODO É CASE SENSITIVE. SE FOR “a”, SÓ SUBSTITUI OS “a” e não os “A”

```
nome = nome.replace("Abella","Souza");
System.out.println(nome);

nome = nome.replace("a","e");
System.out.println(nome);
```

VERIFICAR SE UMA STRING É VAZIA “”

VAZIA, NÃO NULL! SE FOR NULL, SERIA IF(NOME == NULL)

```
boolean ehStringVazia = nome.isEmpty();
System.out.println(ehStringVazia);
```

VERIFICAR SE UMA STRING TERMINA COM DADA STRING

SE TERMINA EXATAMENTE COM A STRING INFORMADA

```
boolean terminaComAbella = nome.endsWith("Abella");
System.out.println(terminaComAbella);
```

VERIFICAR SE UMA STRING INICIA COM DADA STRING

SE INICIA EXATAMENTE COM A STRING INFORMADA

```
boolean iniciaComDaniel = nome.startsWith("Daniel");
System.out.println(iniciaComDaniel);
```

CONVERTER DE STRING PARA INT

CASO VENHA VALOR INVÁLIDO, SERÁ LANÇADA NumberFormatException

```
String numero1Str = "12345";
int numero1 = Integer.parseInt(numero1Str);
System.out.println(numero1);
```

CONVERTER DE STRING PARA DOUBLE

CASO VENHA VALOR INVÁLIDO, SERÁ LANCADA NumberFormatException

```
String numero2Str = "12.1";
double numero2 = Double.parseDouble(numero2Str);
System.out.println(numero2);
```

CONVERTER DE TIPOS PRIMITIVOS PARA STRING (VERSÃO 1)

SE APLICA A TODOS OS TIPOS PRIMITIVOS: INT, DOUBLE, LONG, ETC

```
int numero3 = 12345;
String numero3StrV1 = String.valueOf(numero3);
System.out.println(numero3StrV1);
```

CONVERTER DE TIPOS PRIMITIVOS PARA STRING (VERSÃO 2)

SE APLICA A TODOS OS TIPOS PRIMITIVOS: INT, DOUBLE, LONG, ETC

```
int numero4 = 12345;
String numero4StrV2 = numero4 + "";
System.out.println(numero4StrV2);
```

OBTÉM UM CARACTERE NO ÍNDICE INDICADO

COM BASE NO ÍNDICE INFORMADO, OBTÉM O CARACTERE

```
char charNaPosicao0 = nome.charAt(0);
```

RETIRA OS ESPAÇOS DE UMA STRING

REMOVE ESPAÇOS NO INÍCIO E NO FIM DE UMA STRING

```
String nomeComEspacos = " Daniel    ";
String nomeSemEspacos = nomeComEspacos.trim();
```

VALIDAR STRING COM BASE EM REGEX

REGEX SÃO EXPRESSÕES REGULARES USADAS GERALMENTE PAI
VALIDAR ITENS COMO FORMATO DE CEP, FORMATO DE CELULAR,

```
Pattern p = Pattern.compile("^\\d{9}$");
String telefone = "123456789";
Matcher m = p.matcher(telefone);
boolean telefoneValido = m.matches();
```

+ Sobre Expressões Regulares

Para conhecer mais sobre Expressões Regulares (Regex), peço que consulte:
https://productoversee.com/o-basico-sobre-expressoes-regulares/

OBTER SUBSTRING (VERSÃO 1)

RETORNA UMA NOVA STRING BASEDO NO INTERVALO INFORMADO,
INFORMANDO DE ONDE COMEÇA E ATÉ ONDE DEVE IR. NO EXEMPLO ABAIXO, A NOVA STRING INICIA NA POSIÇÃO 7 E VAI ATÉ A POSIÇÃO 13, POIS NO MÉTODO SUBSTRING INFORMAMOS

```
String nomeCompleto = "Daniel Abella";
String sobrenome = nomeCompleto.substring(7,13);
System.out.println(sobrenome);
```

OBTER SUBSTRING (VERSÃO 2)

MESMO MÉTODO DA VERSÃO 1, PORÉM NÃO PRECISA ESPECIFICAR O INTERVALO. PRECISAMOS ESPECIFICAR APENAS O ÍNDICE DE INÍCIO (INCLUSIVO) E POR PADRÃO, O TÉRMINO É O FIM DA STRING.

```
String nomeCompleto = "Daniel Abella";
String sobrenome = nomeCompleto.substring(7);
System.out.println(sobrenome);
```

CORTANDO UMA STRING COM SPLIT

CORTA UMA STRING COM BASE NO VALOR INFORMADO, CRIANDO UM ARRAY DE STRING COM OS “PEDAÇOS”

```
String nomeCompleto = "Daniel-Abella-Mendonça";
String[] pedacosNome = nomeCompleto.split("-");
//gera um array com 3 Strings:
//["Daniel","Abella",Mendonça"]
```

POR:
DANIEL ABELLA
www.daniel-abella.com