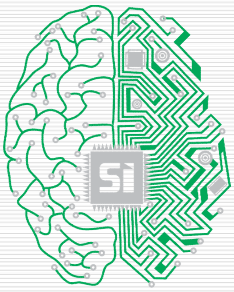
Manipulação de String

Manipulação de variáveis do tipo String



1

Manipulação de String

Diferente de outras linguagens, o Java tem um tratamento para o tipo String totalmente diferenciado.

Isto se deve ao fato de que no Java, uma String tem algumas características de Objeto.

2

Manipulação de String

Um exemplo dessa diferenciação pode ser vista abaixo:

```
String S = "joao";
String S1 = "joao";
if (S == S1){
   System.out.println("São iguais");
}
```

Até a versão 6 do Java, a comparação acima não era verdadeira, em versões atuais isso já é verdadeiro.

3

Manipulação de String

Isso ocorria exatamente pelo fato de String´s não serem variáveis de tipo primitivo.

Nestas situações o que era comparado não era o valor da variável e sim a referência de memória onde o mesmo foi alocado.

Como não existem duas variáveis alocadas na mesma referência, a comparação jamais retornaria verdadeira.

4

Manipulação de String

O Java fornece uma série de métodos para a manipulação do tipo String.

Serão apresentados os principais métodos.

5

Manipulação de String

Todos os métodos que manipulam objetos do tipo String, tem basicamente o mesmo formato.

```
Variável.Ação(<Parâmetros>)
```

Os métodos que manipulam String irão sempre ter retorno e devem ter o retorno tratado de alguma forma.

6

Principais Métodos

Vamos tomar como exemplo para os métodos apresentados, a utilização de uma variável declarada como “S”.

```
String S;
```


Principais Métodos

`length()` – retorna um inteiro, representando o número de caracteres da String (inclusive espaços em branco).

Exemplo:

```
S = “joao”;
System.out.println(S.length()+”caracteres”);
```

Irá retornar o número 4

Principais Métodos

`isEmpty()` – retorna um boolean indicando se a String está Vazia. É uma comparação equivalente ao teste: S.length()==0

Exemplo:

```
If (S.isEmpty()){
   System.out.println("Está VAZIA");
}
```

9

Principais Métodos

`trim()` – retorna uma String sem espaços em branco nos extremos direito e esquerdo, mantendo os espaços do meio. A String original não é alterada.

Exemplo:

```
S = "   joao da silva   ";
System.out.println(S.trim()+" sem espaços");
```

Irá retornar: "joao da silva"

10

Principais Métodos

`charAt(<índice>)` – diferente do Pascal/Delphi onde os caracteres podem ser acessados como um vetor, o Java não permite esse acesso "direto", apesar de seguir o mesmo conceito. String em Java, assim como vetores, tem o seu primeiro índice "0".

Exemplo:

0	1	2	3
J	O	A	O

```
S = "joao";
System.out.println(S.charAt(2)+" é a letra no indice 2");
```

Irá retornar: 'A'

11

Principais Métodos

`replaceALL(<antigo>,<novo>)` – substitui todas as ocorrências de <antigo> pela sequencia definida em <novo>Exemplo:

```
String s= "joao";
s= s.replaceAll("o", "O");
```

Irá retornar: "jOaO"

12

Principais Métodos

substring(<inicio>,<final>) – retorna uma string que é uma substring desta string. A substring iniciano indice especificado em <inicio> e termina em <final-1>. Exemplo:

```
String s= "joaozinho";
String s1=s.substring(3, 5);
```

A variável s1 conterá: “oz”

13

Principais Métodos

Assim como outras linguagens o Java tem funções de conversão para Maiúsculo/Minúsculo.

toUpperCase()- retorna a String com seus caracteres convertidos para maiúsculo, sem alterar a String original.
toLowerCase()- retorna a String com seus caracteres convertidos para minúsculo, sem alterar a String original.

14

Principais Métodos

Exemplo:

```
S = “Joao da Silva”;
System.out.println(“Em
maiusculo”+S.toUpperCase());
System.out.println(“Em
minusculo”+S.toLowerCase());
```

15

Principais Métodos

Segundo o exemplo dado no início da apresentação, a comparação entre String´s não pode ser realizada como em outros tipos, exigindo para isso, métodos para esta ação.

O Java fornece basicamente duas opções de comparação. Comparação de igualdade ou comparação de extremos.

16

Principais Métodos

Comparação de Igualdade – a comparação de igualdade tem retorno boolean `true` se as String´s forem iguais e `false` caso contrário.

- `equals(<String 2>)`
- `equalsIgnoreCase(<String 2>)`

Os dois métodos tem o mesmo objetivo, com a diferença que usando `equalsIgnoreCase`, não há distinção entre maiúsculos e minúsculos.

17

Principais Métodos

Exemplo:

```
if (S.equals(S1))
ou
if (S.equalsIgnoreCase(S1)){
   System.out.println(“Strings iguais”);
}
```

18

Principais Métodos

Comparação de Extremos – nessa comparação há o retorno de um inteiro, representando 3 situações distintas.
0 – as duas String´s são iguais
Valor Negativo – a primeira String é menor
Valor Positivo – a segunda String é menor

- `compareTo(<String 2>)`
- `compareToIgnoreCase(<String 2>)`

19

Principais Métodos

Exemplo:

```
if (S.compareTo(S1))
ou
if (S.compareToIgnoreCase(S1)==0){
   System.out.println("Strings
iguais");
}
```

20

Principais Métodos

Comparação Parcial – nessa comparação há o retorno de um booleano representando as 3 situações similares as existentes em SQL.

Inicia com: startsWith(<String>)
Termina com: endsWith(<String>)
Contém: contains(<String>)

21

Principais Métodos

Exemplo:

```
if (S.contains("A")){
}
```

ou

```
if (S.startsWith("A")){
}
```

ou

```
if (S.endsWith("A")){
}
```

22