FIAP GRADUAÇÃO

Agenda

- Manipulação de Strings
- Escape
- Métodos

String

As Strings na programação representam os textos.

No Java, um texto é um objeto da classe String. As Strings são objetos com comportamentos “especiais” no Java, incluindo uma área específica na memória da JVM.

String - Imutabilidade

Uma característica importante das Strings em Java é que elas são imutáveis. Ou seja, uma vez que for criada, ela permanecerá igual até que seja eliminada da memória.

String - Objetos

Assim como qualquer outra classe, podemos criar instâncias (objetos) de String. Podemos criar Strings da seguintes formas:

```
String nome = new String("Fábio");
String sobrenome = "Miyasato";
```

Dica: NullPointerException

Sempre que trabalhamos com classes (instâncias) estamos sujeitos ao NullPointerException.
Este erro ocorre quando tentamos chamar algum método ou acessar alguma propriedade de uma variável sem nenhum objeto instanciado.

```
String nome = null;
int tamanhoDoNome = nome.length();
```

Escape

Como fazemos para criar uma String de aspas?

```
String nome ="""";
```

O código acima apresenta um erro pois o compilador entende que o segundo caracter de aspas está fechando o primeiro.

Para resolver este caso utilizamos o “escape” de String.

Escape

O caracter \ indica que o caracter seguinte não é um caracter especial (como aspas por exemplo).
Também podemos utilizar o escape para pular linha ao adicionar tabulação horizontal (tab).

Sequência de Escape	Descrição
\t	Move o cursor para a próxima posição da tabulação horizontal
\n	Move o cursor para o começo da próxima linha
\'	Gera a saída do caractere de aspa simples (')
\"	Gera a saída do caractere de aspas duplas (")
\\	Gera a saída do caractere de barra invertida (\)

Concatenação

Podemos concatenar (ou juntar) duas Strings para criar uma terceira String.

```
String nome = "Fabio";
String sobrenome = "Miyasato";

String nomeCompleto = nome + " " + sobrenome;
```

Concatenação

- Podemos concatenar (ou juntar) duas Strings para criar uma terceira String.

```
String nome = "Fabio";
String sobrenome = "Miyasato";

String nomeCompleto = nome + " " + sobrenome;
```

Também existem métodos e classes para concatenar Strings como StringBuilder, método concat() entre outros.

Métodos

Na classe String existem diversos métodos que podemos utilizar:
Métodos para comparação entre Strings (retornam um boolean):

equals(String) -> Se duas String tem exatamente o mesmo valor
equalsIgnoreCase(String) -> Se as Strings têm o mesmo valor ignorando maiusculas e minusculas.
startWith(String) -> Se uma String tem o mesmo início que outra
endWith(String) -> Se uma String tem o mesmo fim que outra

Métodos

Métodos que criam uma nova String com outras características:

toLowerCase() -> Retorna uma String com caracteres minúsculos
toUpperCase() -> Retorna uma String com caracteres maiúsculos
replace(String, String) -> Cria uma nova String trocando “parte” da String por outro valor

Métodos

Já vimos que uma String é na verdade um array (lista) de caracteres. Então podemos pensar nas Strings da seguinte forma:

indice	1	2	3	4
	F	I	A	P

Temos métodos para acessar os caracteres de algum índice ou retornar o indice de determinado caracter

Métodos

length() -> Tamanho da String em int
charAt(int) -> Qual o char na posição?
indexOf(String/Char) -> Qual o indice da String/Char?
subString(int, int) -> Retorna uma nova String entre as posições informadas

OBRIGADO

FIAP