
ORACLE
Academy

Java Foundations

4-3

A Classe String

ORACLE Academy

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Objetivos

- Esta lição abrange os seguintes objetivos:
  - Localizar a classe String na documentação da API Java
  - Entender os métodos da classe String
  - Comparar dois objetos String utilizando léxico
  - Encontrar a localização de uma substring em um objeto String
  - Extrair uma substring de um objeto String

ORACLE Academy

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

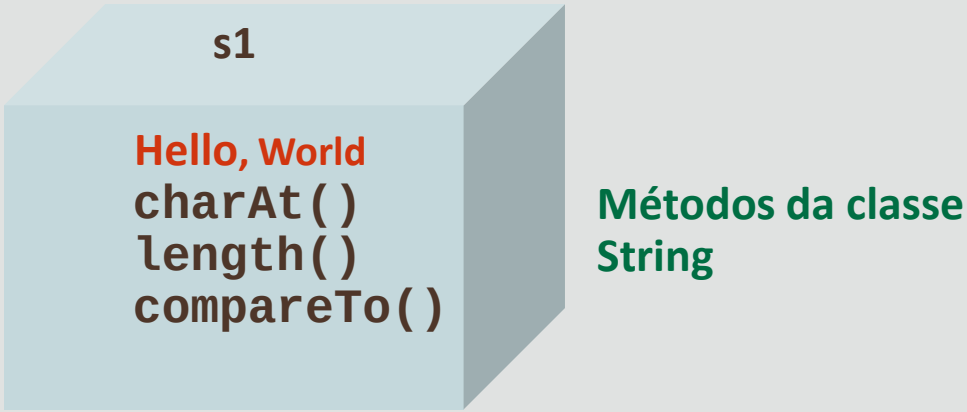
O que É uma String?

- Uma string é uma sequência de caracteres que inclui letras do alfabeto, caracteres especiais e espaço em branco
- Por exemplo:
  - "Como você está?" é uma string que contém letras, espaço em branco e um caractere especial ('?')
- Em Java, as strings não são um tipo de dados primitivo
- Em vez disso, elas são objetos da classe String

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Representando Strings em Java

- Em Java, as strings são objetos da classe denominada java.lang.String
- Exemplo:
  - String s1= “Hello, World”;



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Representando Strings em Java

- Uma string em Java é mais abstrata
- Ou seja, você não precisa conhecer sua estrutura interna, o que facilita seu uso
- Seus métodos permitem que um programador execute operações nela

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Usando a Classe String

- A classe String:
  - É uma das muitas classes incluídas nas bibliotecas de classes Java
  - É parte do java.lang.package
  - Permite a você manter uma sequência de caracteres de dados
- Você usará a classe String frequentemente em todos os seus programas
- Portanto, é importante entender algumas das características especiais das strings em Java

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Documentação da Classe String

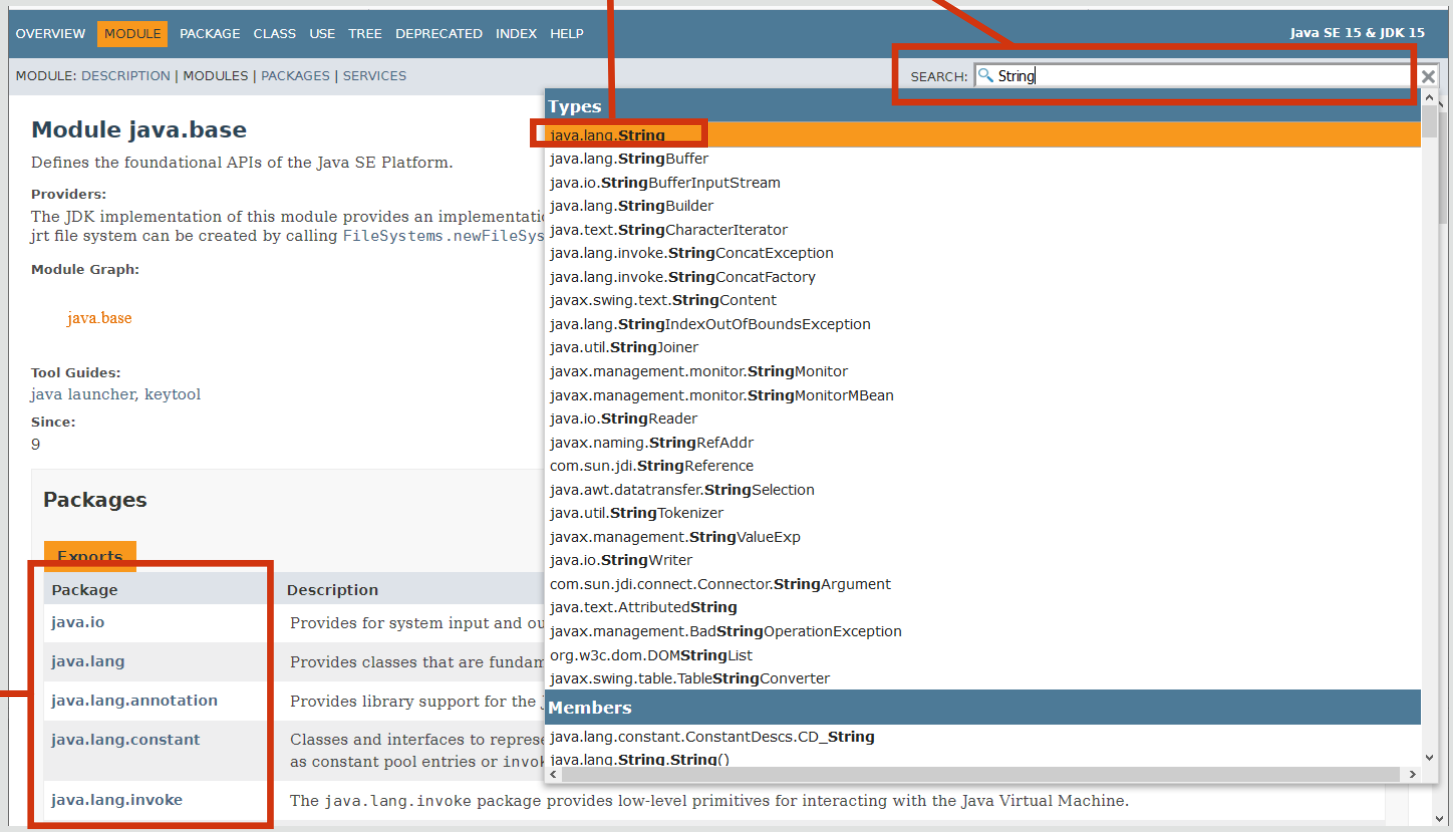
- Você pode acessar a documentação da classe Java String em:
  - https://docs.oracle.com/en/java/javase/17/docs/api/java.base/module-summary.html

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Documentação do Java Platform SE 17 da Classe String

Procure um pacote aqui. Digite String na caixa de pesquisa. Nos Tipos exibidos, selecione java.lang.String

Role a tela para baixo e selecione os pacotes aqui



ORACLE Academy

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

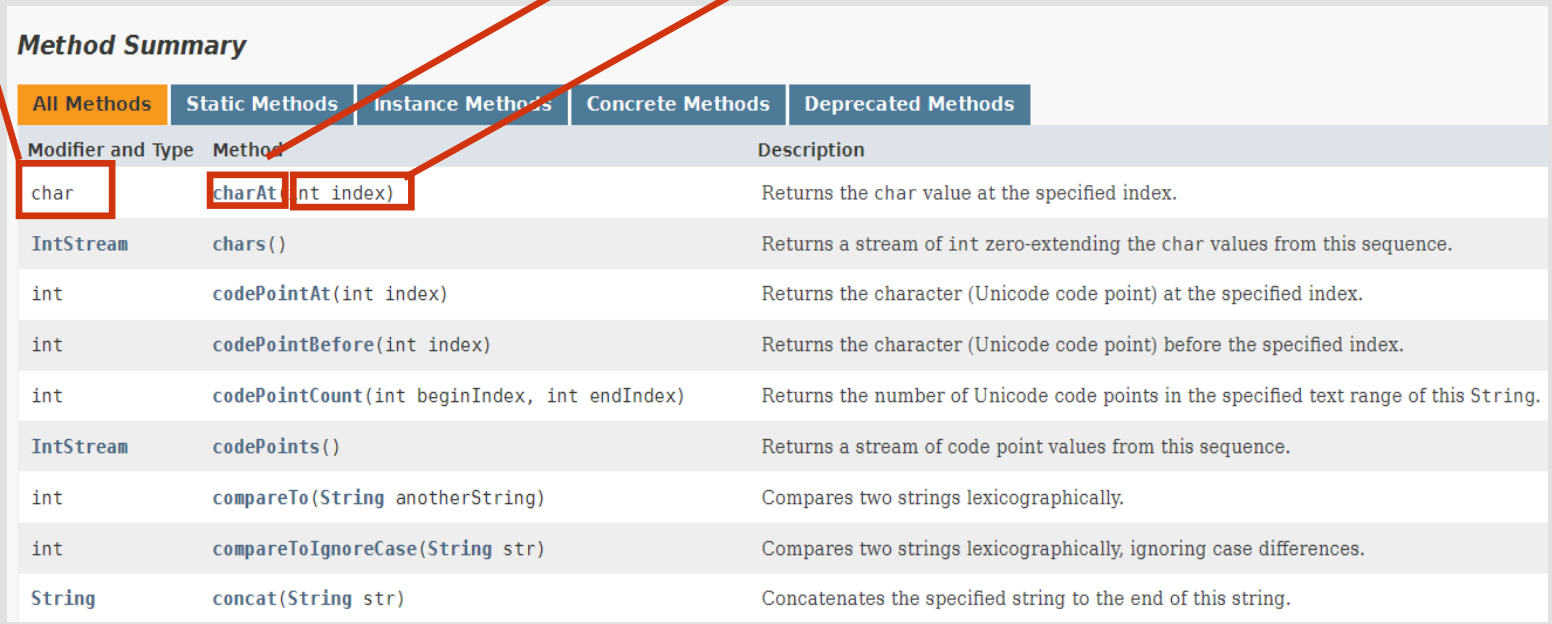
String: Resumo do Método

- `public int charAt(int index)`

Tipo de retorno do método

Nome do método

Tipo de dados do parâmetro que deve ser passado para o método



Method Summary

All Methods | Static Methods | Instance Methods | Concrete Methods | Deprecated Methods

Modifier and Type	Method	Description
char	charAt(int index)	Returns the char value at the specified index.
IntStream	chars()	Returns a stream of int zero-extending the char values from this sequence.
int	codePointAt(int index)	Returns the character (Unicode code point) at the specified index.
int	codePointBefore(int index)	Returns the character (Unicode code point) before the specified index.
int	codePointCount(int beginIndex, int endIndex)	Returns the number of Unicode code points in the specified text range of this String.
IntStream	codePoints()	Returns a stream of code point values from this sequence.
int	compareTo(String anotherString)	Compares two strings lexicographically.
int	compareToIgnoreCase(String str)	Compares two strings lexicographically, ignoring case differences.
String	concat(String str)	Concatenates the specified string to the end of this string.

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

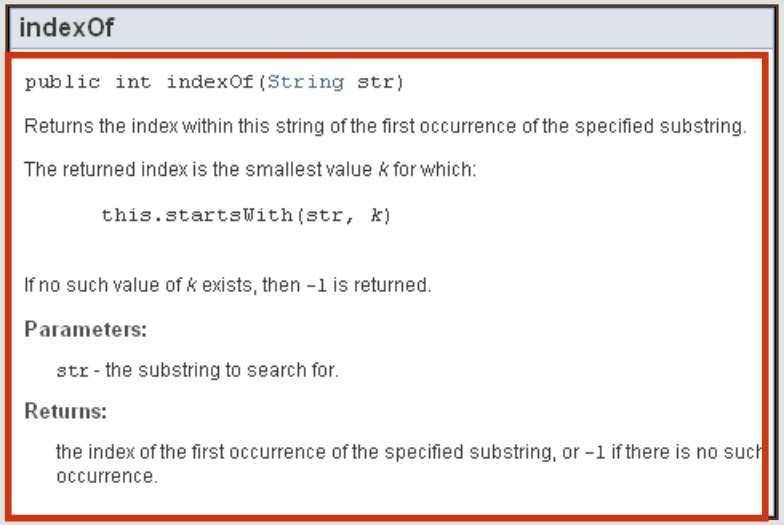
String: Detalhe do Método

Clique aqui para obter uma descrição detalhada do método

```
int     indexOf(String str)
int     indexOf(String str, int fromIndex)
```

Uma descrição detalhada do método indexOf()

Informações adicionais sobre parâmetros e o valor de retorno são mostradas na lista de métodos



indexOf

public int indexOf(String str)

Returns the index within this string of the first occurrence of the specified substring.

The returned index is the smallest value *k* for which:

```
this.startsWith(str, k)
```

If no such value of *k* exists, then -1 is returned.

Parameters:

str - the substring to search for.

Returns:

the index of the first occurrence of the specified substring, or -1 if there is no such occurrence.

ORACLE Academy

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Métodos de String: length

- Você pode calcular o comprimento de uma string usando o método length definido na classe String:
  - Método: name.length()
  - Retorna o comprimento ou o número de caracteres no nome como um valor inteiro
- Exemplo:

```
String name = "Mike.W";

System.out.println(name.length()); //6
```


Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Acessando Cada Caractere em uma String

- Você pode acessar cada caractere em uma string por seu índice numérico
- O primeiro caractere da string está no índice 0, o seguinte está no índice 1 e assim por diante
- Por exemplo:
- `String str= "Hello, World";`

H	e	l	l	o	,		W	o	r	l	d
0	1	2	3	4	5	6	7	8	9	10	11

  - str tem de 0 a 11 índices; ou seja, entre 0 a str.length()-1

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Métodos de String: indexOf()

- Cada caractere de uma string tem um índice
- Você pode recuperar o valor do índice de um caractere na string usando o método indexOf:

Método	Descrição
`str.indexOf(char c)`	Retorna o valor do índice da primeira ocorrência de c na String str
`s1.indexOf(char c, int beginIdx)`	Retorna o valor do índice da primeira ocorrência de c em String s1, começando em beginIdx até o fim da string

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Métodos de String: indexOf()

```
public static void main(String args[]){
    String phoneNum = "404-543-2345";
    int idx1 = phoneNum.indexOf('-’);
    System.out.println("índice do primeiro hífen: "+ idx1); //3
    int idx2 = phoneNum.indexOf('-’, idx1+1);
    System.out.println("índice do segundo hífen: "+ idx2); // 7
}//fim do método main
```


Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Métodos de String: charAt

- Retorna o caractere da string localizada no índice passado como o parâmetro
- Método: str.charAt(int index)

```
String str = "Susan";
System.out.println(str.charAt(0));  //S
System.out.println(str.charAt(3));  //a
```


Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Métodos de String: substring()

- Você pode extrair uma substring de determinada string
- O Java fornece dois métodos para essa operação:

Método	Descrição
`str.substring(int beginIdx)`	Retorna a substring de beginIdx até o fim da string
`str.substring(int beginIdx, int endIdx)`	Retorna a substring de beginIdx até, mas não inclusive, endIdx

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Métodos de String: substring()

```
public static void main(String args[]){
    String greeting = "Hello, World!";
    String sub = greeting.substring(0, 5); → "Hello"
    String w = greeting.substring(7, 11); → "Worl"
    String tail = greeting.substring(7); → "World!"
}//fim do método main
```


Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Métodos de String: replace()

- Este método substitui todas as ocorrências dos caracteres correspondentes em uma string
- Método: replace(char oldChar,char newChar)
- Exemplo:

```
public static void main(String args[]) {
String str = "Usando a String replace para substituir "
             + "caractere";
    String newString = str.replace("r", "R");
    System.out.println(newString);
}//fim do método main
```

  - Saída: Usando a String Replace para Substituir CaRacteRe
  - Todas as ocorrências de um "r" minúsculo são substituídas por um "R" maiúsculo

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Métodos de String: replaceFirst()

- Este método só substitui a 1a. ocorrência do padrão de caracteres correspondentes em uma string
- Método: replaceFirst(String pattern, String replacement)

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Métodos de String: replaceFirst()

- Exemplo:

```
public static void main(String args[]) {
    String replace = "String replace with replaceFirst";
    String newString = replace.replaceFirst("re", "RE");
    System.out.println(newString);
}//fim do método main
```

- Saída:
  - String REplace com replaceFirst
- Só a primeira ocorrência de "re" é substituída por "RE“
- A segunda ocorrência não é alterada

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

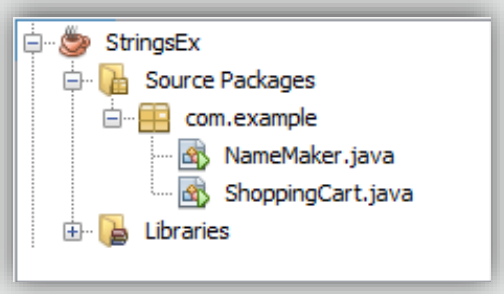
Exercício 1, Parte 1

- Crie um novo projeto e adicione os arquivos `ShoppingCart.java` e `NameMaker.java` a ele
- Examine `ShoppingCart.java`
- Faça o seguinte:
  - Use o método indexOf para obter o índice do caractere de espaço (" ") dentro de custName
  - Atribua-o a spaceIdx
  - Use o método da substring e spaceIdx para obter a parte do primeiro nome de custName
  - Atribua-o a firstName e imprima firstName

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Exercício 1, Parte 2

- Você deve ter percebido que esse projeto tem dois arquivos
- Java com métodos main
  - Isso pode parecer uma contradição porque orientamos a nunca usar mais de um método main



- Às vezes, os programadores fazem isso quando estão testando bits pequenos de código e desejam manter todos os arquivos organizados em um projeto
  - Pressionar Run em seu IDE sempre executa o mesmo arquivo, e nunca os outros
  - Você precisará clicar com o botão direito do mouse no outro arquivo que deseja executar Aparecerá um menu com uma opção para executar esse arquivo

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Declarando e Criando uma String

- Você pode instanciar strings de duas maneiras:
- Literais de string:
  - Atribua diretamente uma literal de string a uma referência de string


```
String hisName = "Fred Smith";
```

- Operador new:
  - Semelhante a qualquer outra classe
  - Não é usado comumente nem é recomendado

```
String herName = new String("Anne Smith");
```

A palavra-chave new

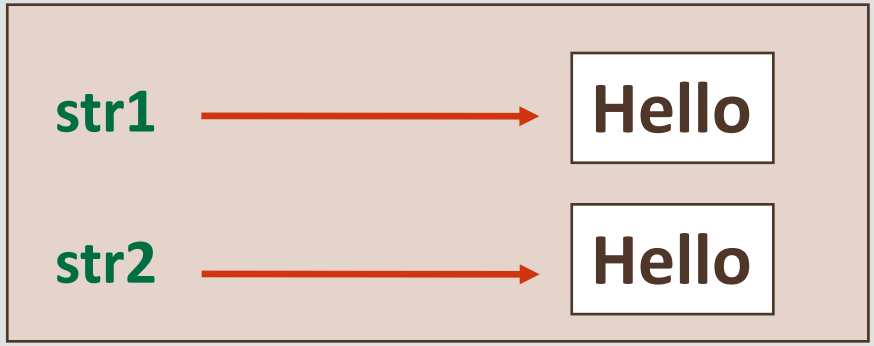
Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

As Strings São Imutáveis

- Um objeto String é imutável; ou seja, depois que um objeto String é criado, seu valor não pode ser alterado
- Como as strings são imutáveis, o Java pode processá-las de maneira muito eficiente
  - Considere o seguinte:

```
String str1 = "Hello";
String str2 = "Hello";
```

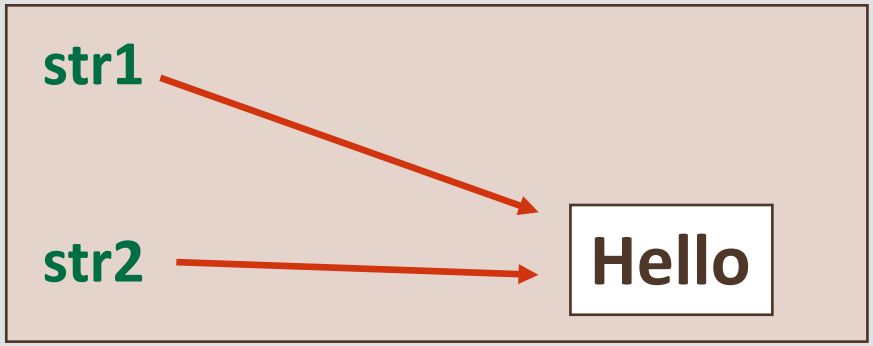
  - Esperamos isso...



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

As Strings São Imutáveis

- Mas isso é o que acontece…



- O sistema de run-time Java sabe que as duas strings são idênticas e aloca o mesmo local de memória para os dois objetos

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

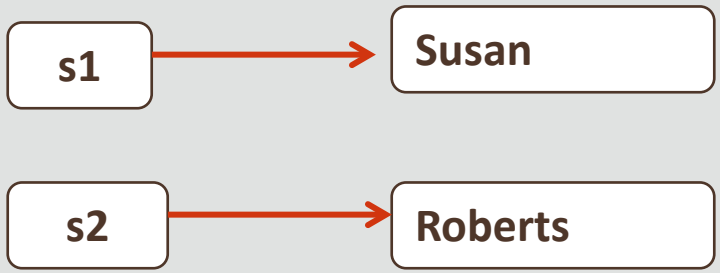
Concatenando Strings

- No Java, a concatenação de strings forma uma nova string que é a combinação de várias strings
- Você pode concatenar strings em Java de duas maneiras:
  - operador de concatenação de strings **+**
  - método **concat()**

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Usando o Operador + (Antes da Concatenação)

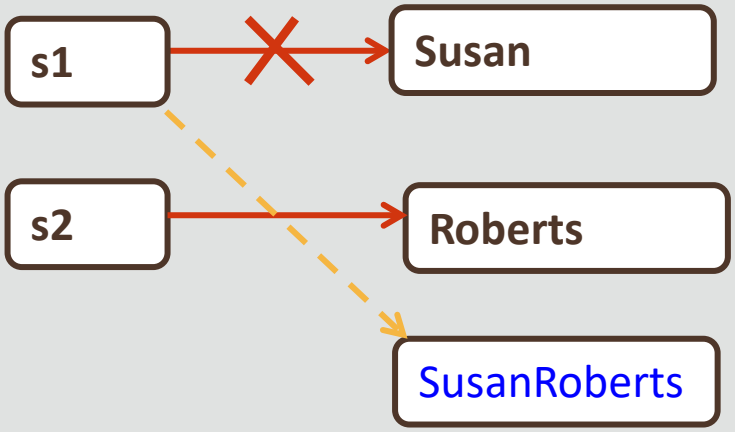
```
public static void main(String args[]) {
    String s1 = "Susan";
    String s2 = "Roberts";
}//fim do método main
```



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Usando o Operador + (Depois da Concatenação)

```
public static void main(String args[]) {
    String s1 = "Susan";
    String s2 = "Roberts";
    S1 = s1 + s2;
    System.out.println(s1);
}//fim do método main
```



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Concatenando Dados Não String com String

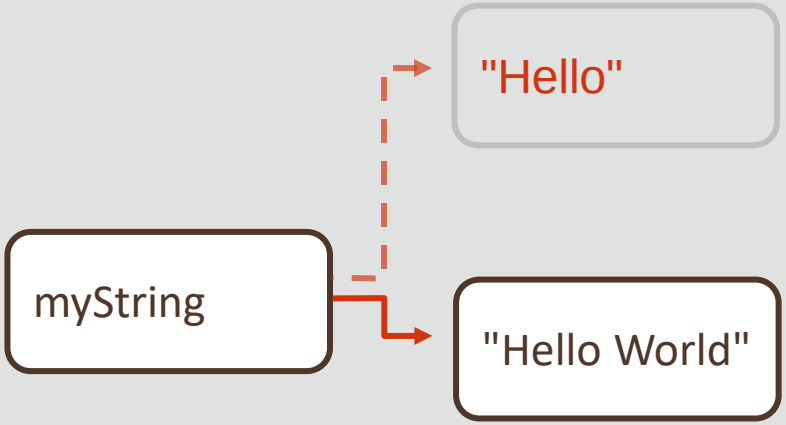
- Se um dos operandos for uma string, o Java converterá os tipos de dados não string automaticamente em strings antes da concatenação
- Exemplo:

```
public static void main(String args[]) {
    String newString = "Learning Java" + 17;
    System.out.println(newString);                  //Learning Java 17

    System.out.println("Total : " + 17 + 17);    //Total: 1717
    System.out.println("Total : " + (17 + 17)); //Total: 34

    String numString1 = "17" + 17;
    System.out.println(numString1);                 //1717
}//fim do método main
```


Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Usando o Método concat() (Antes da Concatenação)

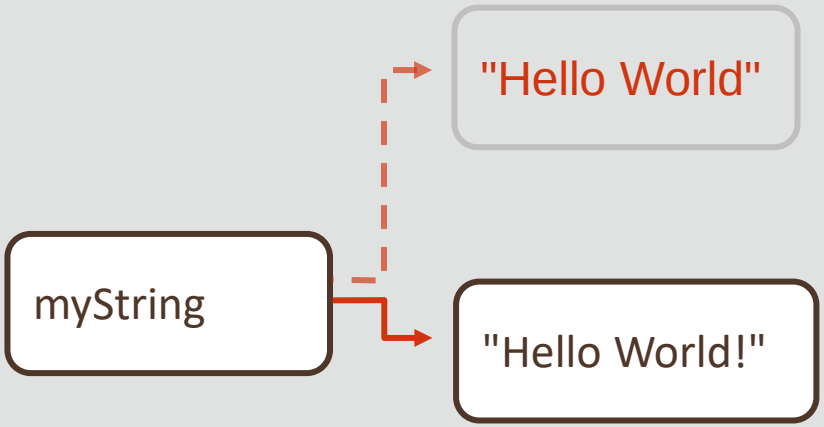
```
String myString = "Hello";
myString = myString.concat(" World");
```



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Usando o Método concat() (Após a Concatenação)

```
String myString = "Hello";
myString = myString.concat(" World");
myString = myString + "!"
```



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Exercício 2

- Abra o projeto que você criou no Exercício 1
- Examine `NameMaker.java`
- Faça o seguinte:
  - Declare variáveis de String: firstName, middleName, lastName e fullName
  - Solicite que usuários insiram os respectivos nomes, nomes do meio e sobrenomes e leiam os nomes no teclado
  - Defina e exiba fullName como firstName+a espaço em branco char+middleName+a espaço em branco char+lastName

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Exercício 2

- O que você acha que é preferível para este cenário?
- Ou seja, o operador de concatenação de string ou o método concat()?

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

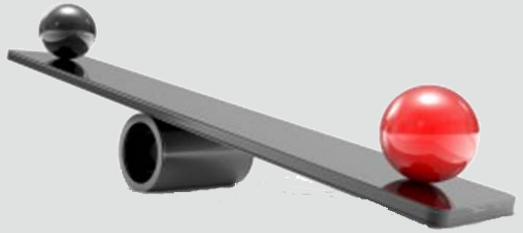
Qual é a Maneira Preferida para Concatenar Strings?

- Como você observou no exercício anterior:
- operador +:
  - Pode funcionar entre uma string e uma string ou um valor de tipo de dados char, int, double ou float
  - Converte o valor em sua representação de string antes da concatenação
- Método concat():
  - Só pode ser chamado em strings
  - Verifica a compatibilidade dos tipos de dados Será produzido um erro de tempo de compilação se não houver compatibilidade

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Como Você Compara Objetos String?

- Você pode comparar dois objetos String usando o método compareTo()
- Esse método compara com base na ordem lexicográfica das strings
- As comparações lexicográficas são semelhantes à ordenação encontrada em um dicionário
- As strings são comparadas caractere por caractere até sua ordem ser determinada ou até provarem ser idênticas
- Sintaxe: **s1.compareTo(s2)**
- Retorna um valor inteiro que indica a ordem das duas strings



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Valor Retornado por compareTo()

- O valor inteiro retornado pelo método compareTo() pode ser interpretado da seguinte maneira:
  - Retorna < 0 quando a string que está chamando o método é a primeira lexicograficamente
  - Retorna == 0 quando as duas strings são lexicograficamente equivalentes
  - Retorna > 0 quando o parâmetro passado para o método é o primeiro lexicograficamente

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Usando o Método compareTo()

- Vamos analisar alguns exemplos:
  - "computer".compareTo("comparison")
    - Retorna um valor inteiro > 0 porque o parâmetro "comparison" é o primeiro lexicograficamente
  - "cab".compareTo("car")
    - Retorna um valor inteiro < 0 porque a string “cab” que está chamando o método é a primeira lexicograficamente
  - "car".compareTo("car")
    - Retorna um valor inteiro igual a 0 porque ambos são lexicograficamente equivalentes

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Usando o Método compareTo(): Exemplo

- Vamos escrever um programa para comparar nomes usando o método compareTo():

```
public static void main(String[] args) {

    String s1 = "Susan";
    String s2 = "Susan";
    String s3 = "Robert";

    //Retorna 0 porque s1 é idêntico a s2
    System.out.println(s1.compareTo(s2)); //Output is 0

    //Retorna > 0 porque 'S' vem depois de 'R'
    System.out.println(s1.compareTo(s3)); // Output is 1

    //Retorna < 0 porque 'R' vem antes de 'S'
    System.out.println(s3.compareTo(s1)); // Output is -1
}//fim do método main
```


Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Resumo

- Nesta lição, você deverá ter aprendido a:
  - Localizar a classe String na documentação da API Java
  - Entender os métodos da classe String
  - Comparar dois objetos String lexicograficamente
  - Encontrar a localização de uma substring em um objeto String
  - Extrair uma substring de um objeto String

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.


ORACLE
Academy