**Java**

***Strings***

String related classes

* Java provides three String related classes
* java.lang package

– ***String*** class: Storing and processing Strings but Strings created using the String class cannot be modified (**immutable**)

– ***StringBuffer*** class: Create flexible Strings that can be modified

* java.util package

– ***StringTokenizer*** class: Can be used to extract tokens from a String

Prepared by - Rifat Shahriyar 2

***String***

Prepared by - Rifat Shahriyar 3

String

* String class provide many constructors and more than 40 methods for examining in individual characters in a sequence
* You can create a String from a String value or from an array of characters.

– String newString = new String(stringValue);

* The argument stringValue is a sequence of characters enclosed inside double quotes

– String message = new String (“Welcome”);

– String message = “Welcome”;

Prepared by - Rifat Shahriyar 4

String Constructors



Prepared by - Rifat Shahriyar 5

String Length

* Returns the length of a String

– ***length()***

* Example:

***String s1=“Hello”;***

***System.out.println(s1.length());***

Prepared by - Rifat Shahriyar 6

Extraction

* Get the character at a specific location in a string

– ***s1.charAt(1)***

* Get the entire set of characters in a string

– ***s1.getChars(0, 5, charArray, 0)***

Prepared by - Rifat Shahriyar 7

Extracting Substrings

* substring method enable a new String object to be created by copying part of an existing String object

– ***substring (int startIndex)*** ‐ copies the characters form the starting index to the end of the String

– ***substring(int beginIndex, int endIndex)*** ‐ copies the characters from the starting index to one beyond the endIndex

Prepared by - Rifat Shahriyar 8

String Comparisons

* *equals*

– Compare any two string objects for equality using lexicographical comparison. ***s1.equals(“hello”)***

* *equalsIgnoreCase*

– ***s1.equalsIgnoreCase(s2)***

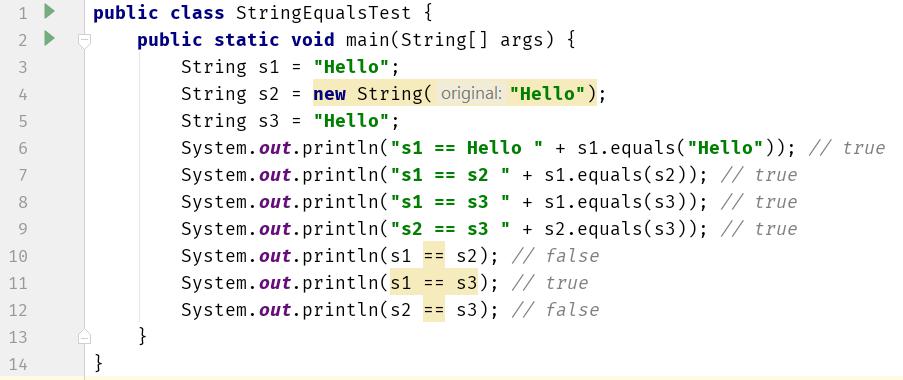
* *compareTo*

– ***s1.compareTo(s2)***

– s1 > s2 (positive), s1 < s2 (negative), s1 = s2 (zero)

Prepared by - Rifat Shahriyar 9

String Comparisons



Prepared by - Rifat Shahriyar 10

String Comparisons

* *regionMatches* compares portions of two Stringobjects for equality

– ***s1.regionMatches (0, s2, 0, 5)***

– ***s1.regionMatches (true, 0, s2, 0, 5)***

* If the first argument is true, the method ignores the case of the characters being compared
* *startsWith* and endsWith check whether a Stringstarts or ends with a specified String

– ***s1.startsWith (s2)***

– ***s1.endsWith (s2)***

Prepared by - Rifat Shahriyar 11

String Concatenation

* Java provide the *concat* method to concatenate two strings.

***String s1 = new String (“Happy ”);***

***String s2 = new String (“Birthday”);***

***String s3 = s1.concat(s2);***

s3 will be “Happy Birthday”

Prepared by - Rifat Shahriyar 12

String Search

* Find the position of character/String within a String

– ***int indexOf(char ch)***

– ***int lastIndexOf(char ch)***

Prepared by - Rifat Shahriyar 13

String Split

* split() method splits a String against given regular expression and returns a character array
* ***String test = "abc,def,123";***

***String[] out = test.split(",");*** out[0] - abc, out[1] - def, out[2] - 123

Prepared by - Rifat Shahriyar 14

String Conversions

* Generally, the contents of a String cannot be changed once the string is created,
* Java provides conversion methods
* ***toUpperCase()*** and ***toLowerCase()***

– Converts all the characters in the string to lowercase or uppercase

* ***trim()***

– Eliminates blank characters from both ends of the string

* ***replace(oldChar, newChar)***

– Replaces a character in the string with a new character

Prepared by - Rifat Shahriyar 15

String to Other Conversions

* The String class provides ***valueOf*** methods for converting a character, an array of characters and numeric values to strings

– ***valueOf*** method take different argument types

Prepared by - Rifat Shahriyar 16

String to Other Conversions

|  |  |  |
| --- | --- | --- |
| **Type** | **To String** | **From String** |
|  |  |  |
| boolean | String.valueOf(boolean) | Boolean.parseBoolean(String) |
|  |  |  |
| byte | String.valueOf(int) | Byte.parseByte(String, int base) |
|  |  |  |
| short | String.valueOf(int) | Short.parseShort (String, int base) |
|  |  |  |
| Int | String.valueOf(int) | Integer.parseInt (String, int base) |
|  |  |  |
| long | String.valueOf(long) | Long.parseLong (String, int base) |
|  |  |  |
| float | String.valueOf(float) | Float.parseFloat(String) |
|  |  |  |
| double | String.valueOf(double) | Double.parseDouble(String) |
|  |  |  |

Prepared by - Rifat Shahriyar 17

String Conversion Example

* To convert an int to a String (3 different ways):

***int n = 123;***

***String s1 = Integer.toString(n);***

***String s2 = String.valueOf(n);***

***String s3 = n + "";***

* To convert a string to an int:

***String s = “1234”;***

***int n = Integer.parseInt(s);***

Prepared by - Rifat Shahriyar 18

***StringBuffer***

Prepared by - Rifat Shahriyar 19

StringBuffer

* Can be used wherever a string is used

– More flexible than String

– Can add, insert, or append new contents into a string buffer

* The StringBuffer class has three constructors and more than 30 methods for managing the buffer and for modifying strings in the buffer
* Every StringBuffer is capable of storing a number of characters specified by its capacity

Prepared by - Rifat Shahriyar 20

StringBuffer Constructors

* ***public StringBuffer()***

– No characters in it and an initial capacity of 16 characters

* ***public StringBuffer(int length)***

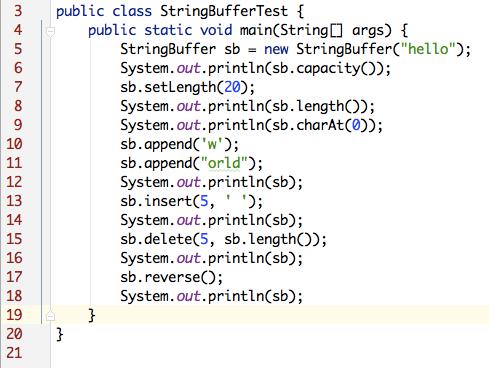
– No characters in it and an initial capacity specified by the length argument

* ***public StringBuffer(String string)***

– Contains String argument and an initial capacity of the buffer is 16 plus the length of the argument

Prepared by - Rifat Shahriyar 21

StringBuffer



Prepared by - Rifat Shahriyar 22

***StringTokenizer***

Prepared by - Rifat Shahriyar 23

StringTokenizer

* Break a string into pieces (tokens) so that information contained in it can be retrieved and processed
* Specify a set of characters as delimiters when constructing a StringTokenizer object
* **StringTokenizer** class is available since **JDK 1.0** andthe **String.split()** is available since **JDK 1.4**
* *StringTokenizer is a legacy class, retained for compatibility reasons, the use is discouraged!*

Prepared by - Rifat Shahriyar 24

StringTokenizer

* Constructors

– ***StringTokenizer(String str, String delim)***

– ***StringTokenizer(String str)***

* Methods

– ***hasMoreToken()*** ‐ Returns true if there is a token left in the string

– ***nextToken()*** ‐ Returns the next token in the string

– ***nextToken(String delim)*** ‐ Returns the next token in the string after reseting the delimiter to delim

– ***countToken( )*** ‐ Returns the number of tokens remaining in the string tokenizer

Prepared by - Rifat Shahriyar 25

StringTokenizer



Prepared by - Rifat Shahriyar 26