**String Handling**

* In java, a string handling is defined as a sequence of characters.
* But unlike , many other languages that implements strings as character arrays , java implements strings as a objects od type String .
* Java handles String by two classes StringBuffer and String . The String and StringBuffer classes are defined in java.lang .
* Thus , they are available to all programs automatically .

1. **String Concatenation(+)**

EX –

import java.util.\*;

class str3{

public static void main(String args[]){

String s1 = new String("java");

String s2 = "2ali";

String s3 = s1+s2;

System.out.println("s1="+s1);

System.out.println("s2="+s2);

System.out.println("Concantenation operator="s1+s2);

System.out.println("s3="+s3);

byte num[]=(56,45,10,70);

String s4=new String(num);

System.out.println("s4="+s4);

}

}

Output :

s1 = Java

s2 = 2all

Concatenation Operator = Java2all

S3 = Java2all

S4 = ABCD

1. **Character Extraction :**

The String class provides way in which characters can be extracted from a string object .

|  |  |
| --- | --- |
| Method | Description |
| Char charAt(int indexNum) | CharAt() function is used to extract a single character from a string. IndexNum is the index number of the character that we want to extract . |
| Void getChars(int sourceStart , int sourceEnd , char target() , int target(Start)) | Used to extract more than one character at a time . sourceStart specifies beginning of the string , and sourceEnd specifies end of the String . The array that will receive the character is specified by the target . The index within target at which the substring will be copied . |
| Byte[] getBytes() | This is an alternative to getChars() that stores the characters in an array of bytes.It uses the default character to byte conversions provided by the platforms . |
| Char[] toCharArray() | Same as getChars |

1. **String Comparision :**

* The String class provides several methods that compare strings or substrings within strings .
* equals() – used to compare two strings General form :
* Boolean equals(Object str)
* Here , str is a String object .
* It returns true if the strings contain the same character otherwise it returns false.
* The comparision is case – sensitive .
* equalsIgnoreCase() – same as equals but this ignores case. Generals form :
* Boolean equalsIgnoreCase(String str)
* Here , str is a string object .
* It returns true if the strings contain the same character otherwise it returns false .
* Boolean regionMatches(Boolean ignoreCase , int startIndex , String str2 , int st2StartIndex , int numChars)

startsWith() and endsWith()

The startsWith() method determines whether a given string begins with a specified string .

endsWith() determines whether the string ends with a specified string .

General form :

Boolean startsWith(String str)

Boolean startsWith(String str)

Equals() Versus ==

Equals() method and the == operator perform two different operations.

The equals() method compares the characters inside a string object . The == operator compares two object references to see whether they refer to the same instances .

CompareTo()

It is enough to know that two strings just for equal or not . For sorting applications , we need to know which is less than , equal to , or greater than the other string.

|  |  |
| --- | --- |
| Method | Description |
| String substring(int n) | Gives substring starting from nth character. |
| String substring (int n , int m) | Given substring starting from nth char up to mth |
| S1.concate(s2) | Concatenates s1 and s2 |
| String replace(char original , char replacement)  String trim() | The replace() method replaces all occurances of one character in the involving string with another character .  Remove white space at the beginning and end of the string |

1. **valueOf() :**

The valueOf() method converts data from internal format into a human - readable form . It has several forms :

String valueOf(double num)

String valueOf(long num)

String valueOf(Object ob)

String valueOf(char chars[])

String valueOf(char chars[], int startIndex , int numChars)

**Methods & Examples**

|  |  |
| --- | --- |
| **Method call** | **Description** |
| S2=s1.toLowerCase; | Converts the string s1 to lowercase |
| S2=s1.toUpperCase; | Converts the string s1 to uppercase |
| S2=s1.replace(‘x’,’y’) | Replace all apperarances of x and y |
| S2=s1.trim() | Remove white spaces at the beginning and of the string s1 |
| S1.equals(s2) | Returns true if s1 and s2 are equals |
| S1.equalsIgnoreCase(s2) | Return true if s1=s2 , ignoring the case of characters |
| S1.length() | Gives the length of s1 |
| S1.CharAt(n) | Gives the nth character of s1 |
| S1.compareTo(s2) | Returns -ve if s1<s2 +ve . If s1>s2 , and 0 if s1=s2 |
| S1.concat(s2) | Concatenates s1 and s2 |
| S1.subString(n) | Gives substring starting from nth character |
| S1.subString(n , m) | Gives substring starting from nth char up to mth |
| String.valueOf(p) | Returns the string representation of the specified type of argument. |
| toString() | This object  (which is already a string ) is itself written |
| S1.IndexOf(‘x’) | Gives the position of first occurrence of ‘x’ in the string s1 |
| S1.IndexOf(‘x’ , n) | Gives the position of ‘x’ that occurs after nth position in the string s1 |
| String.ValuesOf(Variable) | Converts the parameters value of string represents |