**STRING**

**A string is a sequence of character in Java, widely used as an object.**

**char[] ch={'j','a','v','a','t','p','o','i','n','t'};**

**String s=new String(ch);**

**is same as:**

**String s="javatpoint";**

**The CharSequence interface is used to represent the sequence of characters. String, StringBuffer and StringBuilder classes implement it. It means, we can create strings in java by using these three classes.**

****

**The Java String is immutable which means it cannot be changed. Whenever we change any string, a new instance is created. For mutable strings, you can use StringBuffer and StringBuilder classes.**

**What is String in java**

**Generally, String is a sequence of characters. But in Java, string is an object that represents a sequence of characters. The java.lang.String class is used to create a string object.**

**How to create a string object?**

**There are two ways to create String object:**

* **By string literal**
* **By new keyword**

**1) String Literal**

**String s="welcome";**

**Each time you create a string literal, the JVM checks the "string constant pool" first. If the string already exists in the pool, a reference to the pooled instance is returned. If the string doesn't exist in the pool, a new string instance is created and placed in the pool. For example:**

**String s1="Welcome";**

**String s2="Welcome";//It doesn't create a new instance**

**Note: String objects are stored in a special memory area known as the "string constant pool".**

**Why Java uses the concept of String literal?**

**To make Java more memory efficient (because no new objects are created if it exists already in the string constant pool).**

**2) By new keyword**

**String s=new String("Welcome");//creates two objects and one reference variable**

**Java String Example**

**public class StringExample{**

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

**String s1="java";//creating string by java string literal**

**char ch[]={'s','t','r','i','n','g','s'};**

**String s2=new String(ch);//converting char array to string**

**String s3=new String("example");//creating java string by new keyword**

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

**System.out.println(s2);**

**System.out.println(s3);**

**}}**

**Output:**

**java**

**strings**

**example**

**Java String class methods**

**The java.lang.String class provides many useful methods to perform operations on sequence of char values.**

|  |  |  |
| --- | --- | --- |
| **No.** | **Method** | **Description** |
| **1** | [**char charAt(int index)**](https://www.javatpoint.com/java-string-charat) | **returns char value for the particular index** |
| **2** | [**int length()**](https://www.javatpoint.com/java-string-length) | **returns string length** |
| **3** | [**static String format(String format, Object... args)**](https://www.javatpoint.com/java-string-format) | **returns a formatted string.** |
| **4** | [**static String format(Locale l, String format, Object... args)**](https://www.javatpoint.com/java-string-format) | **returns formatted string with given locale.** |
| **5** | [**String substring(int beginIndex)**](https://www.javatpoint.com/java-string-substring) | **returns substring for given begin index.** |
| **6** | [**String substring(int beginIndex, int endIndex)**](https://www.javatpoint.com/java-string-substring) | **returns substring for given begin index and end index.** |
| **7** | [**boolean contains(CharSequence s)**](https://www.javatpoint.com/java-string-contains) | **returns true or false after matching the sequence of char value.** |
| **8** | [**static String join(CharSequence delimiter, CharSequence... elements)**](https://www.javatpoint.com/java-string-join) | **returns a joined string.** |
| **9** | [**static String join(CharSequence delimiter, Iterable<? extends CharSequence> elements)**](https://www.javatpoint.com/java-string-join) | **returns a joined string.** |
| **10** | [**boolean equals(Object another)**](https://www.javatpoint.com/java-string-equals) | **checks the equality of string with the given object.** |
| **11** | [**boolean isEmpty()**](https://www.javatpoint.com/java-string-isempty) | **checks if string is empty.** |
| **12** | [**String concat(String str)**](https://www.javatpoint.com/java-string-concat) | **concatenates the specified string.** |
| **13** | [**String replace(char old, char new)**](https://www.javatpoint.com/java-string-replace) | **replaces all occurrences of the specified char value.** |
| **14** | [**String replace(CharSequence old, CharSequence new)**](https://www.javatpoint.com/java-string-replace) | **replaces all occurrences of the specified CharSequence.** |
| **15** | [**static String equalsIgnoreCase(String another)**](https://www.javatpoint.com/java-string-equalsignorecase) | **compares another string. It doesn't check case.** |
| **16** | [**String[] split(String regex)**](https://www.javatpoint.com/java-string-split) | **returns a split string matching regex.** |
| **17** | [**String[] split(String regex, int limit)**](https://www.javatpoint.com/java-string-split) | **returns a split string matching regex and limit.** |
| **18** | [**String intern()**](https://www.javatpoint.com/java-string-intern) | **returns an interned string.** |
| **19** | [**int indexOf(int ch)**](https://www.javatpoint.com/java-string-indexof) | **returns the specified char value index.** |
| **20** | [**int indexOf(int ch, int fromIndex)**](https://www.javatpoint.com/java-string-indexof) | **returns the specified char value index starting with given index.** |
| **21** | [**int indexOf(String substring)**](https://www.javatpoint.com/java-string-indexof) | **returns the specified substring index.** |
| **22** | [**int indexOf(String substring, int fromIndex)**](https://www.javatpoint.com/java-string-indexof) | **returns the specified substring index starting with given index.** |
| **23** | [**String toLowerCase()**](https://www.javatpoint.com/java-string-tolowercase) | **returns a string in lowercase.** |
| **24** | [**String toLowerCase(Locale l)**](https://www.javatpoint.com/java-string-tolowercase) | **returns a string in lowercase using specified locale.** |
| **25** | [**String toUpperCase()**](https://www.javatpoint.com/java-string-touppercase) | **returns a string in uppercase.** |
| **26** | [**String toUpperCase(Locale l)**](https://www.javatpoint.com/java-string-touppercase) | **returns a string in uppercase using specified locale.** |
| **27** | [**String trim()**](https://www.javatpoint.com/java-string-trim) | **removes beginning and ending spaces of this string.** |
| **28** | [**static String valueOf(int value)**](https://www.javatpoint.com/java-string-valueof) | **converts given type into string. It is an overloaded method.** |

**Immutable String in Java**

**In java, string objects are immutable. Immutable simply means unmodifiable or unchangeable.**

**Once string object is created its data or state can't be changed but a new string object is created.**

**Example:**

**class Testimmutablestring{**

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

**String s="Sachin";**

**s.concat(" Tendulkar");//concat() method appends the string at the end**

**System.out.println(s);//will print Sachin because strings are immutable objects**

**}**

**}**

**Output:**

**Sachin**

**Example:**

**class Testimmutablestring1{**

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

**String s="Sachin";**

**s=s.concat(" Tendulkar");**

**System.out.println(s);**

**}**

**}**

**Output:**

**Sachin Tendulkar**

1. **Java String length():**

**public class Example{**

**public static void main(String args[]{**

**String s1="hello";**

**String s2="whatsup";**

**System.out.println("string length is: "+s1.length());**

**System.out.println("string length is: "+s2.length());**

**}}**

**Here, String length()  function will return the length 5 for s1 and 7 for s2 respectively.**

1. **Java String compareTo():**

**public class CompareToExample{**

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

**String s1="hello";**

**String s2="hello";**

**String s3="hemlo";**

**String s4="flag";**

**System.out.println(s1.compareTo(s2)); // 0 because both are equal**

**System.out.println(s1.compareTo(s3)); //-1 because "l" is only one time lower than "m"**

**System.out.println(s1.compareTo(s4)); // 2 because "h" is 2 times greater than "f"**

**}}**

**This program shows the comparison between the various string. It is noticed that**

**if s1 > s2, it returns a positive number**

**if s1 < s2, it returns a negative number**

**if s1 == s2, it returns 0**

1. **Java String concat() :**

**public class ConcatExample{**

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

**String s1="hello";**

**s1=s1.concat("how are you");**

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

**}}**

**The above code returns “hellohow are you”.**

1. **Java String charAt() :**

**public class CharAtExample{**

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

**String name="javatpoint";**

**char ch=name.charAt(4);//returns the char value at the 4th index**

**System.out.println(ch);**

**}}**

**The above code returns t.**

1. **Java String format() :**

**public class FormatExample{**

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

**String name="sonoo";**

**String sf1=String.format("name is %s",name);**

**String sf2=String.format("value is %f",32.33434);**

**String sf3=String.format("value is %32.12f",32.33434);//returns 12 char fractional part filling with 0**

**System.out.println(sf1);**

**System.out.println(sf2);**

**System.out.println(sf3);**

**}}**

**Output:**

**name is sonoo**

**value is 32.334340**

**value is 32.334340000000**

1. **Java String substring():**

**public class SubstringExample{**

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

**String s1="javatpoint";**

**System.out.println(s1.substring(2,4));//returns va**

**System.out.println(s1.substring(2));//returns vatpoint**

**}}**

**Output:**

**va**

**vatpoint**

1. **Java String contains() :**

**class ContainsExample{**

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

**String name=" hello how are you doing?";**

**System.out.println(name.contains("how are you")); // returns true**

**System.out.println(name.contains("hello")); // returns true**

**System.out.println(name.contains("fine")); // returns false**

**}}**

**In the above code, the first two statements will return true as it matches the String whereas the second print statement will return false because the characters are not present in the string.**

1. **Java String join() :**

**public class StringJoinExample{**

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

**String joinString1=String.join("-","welcome","to","javatpoint");**

**System.out.println(joinString1);**

**}}**

**welcome-to-javatpoint**

1. **Java String equals() :**

**public class EqualsExample{**

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

**String s1="hello";**

**String s2="hello";**

**String s3="hi";**

**System.out.println(s1.equalsIgnoreCase(s2)); // returns true**

**System.out.println(s1.equalsIgnoreCase(s3)); // returns false**

**}**

**}**

1. **Java String IsEmpty() :**

|  |  |  |
| --- | --- | --- |
|  | **public class IsEmptyExample{**  **public static void main(String args[]){**  **String s1="";**  **String s2="hello";**  **System.out.println(s1.isEmpty());      // true**  **System.out.println(s2.isEmpty());      // false**  **}}**   1. **Java String replace():**   **Example 1:**  **public class ReplaceExample1{**  **public static void main(String args[]){**  **String s1="hello how are you";**  **String replaceString=s1.replace('h','t');**  **System.out.println(replaceString);**  **}**  **}**  **In the above code, it will replace all the occurrences of ‘h’ to ‘t’. Output to the above code will be “tello tow are you”.**  **Example 2:**   |  | | --- | | **public class ReplaceExample2{**  **public static void main(String args[]){**  **String s1="Hey, welcome to Edureka";**  **String replaceString=s1.replace("Edureka","Brainforce");**  **System.out.println(replaceString);**  **}}** |   **In the above code, it will replace all occurrences of “Edureka” to “Brainforce”. Therefore, the output would be “ Hey, welcome to Brainforce”.**   1. **Java String equalsIgnoreCase():**   **public class EqualsIgnoreCaseExample{**  **public static void main(String args[]){**  **String s1="hello";**  **String s2="HELLO";**  **String s3="hi";**  **System.out.println(s1.equalsIgnoreCase(s2)); // returns true**  **System.out.println(s1.equalsIgnoreCase(s3)); // returns false**  **}}**  **In the above code, the first statement will return true because the content is same irrespective of the case. Then, in the second print statement will return false as the content doesn’t match in the respective strings.**   1. **Java String split():**   **public class SplitExample{**  **public static void main(String args[]){**  **String s1="java string split method by javatpoint";**  **String[] words=s1.split("\\s");//splits the string based on whitespace**  **//using java foreach loop to print elements of string array**  **for(String w:words){**  **System.out.println(w);**  **}**  **}}**  **Output:**  **java**  **string**  **split**  **method**  **by**  **javatpoint**   1. **Java String intern():**   **public class InternExample{**  **public static void main(String args[]){**  **String s1=new String("hello");**  **String s2="hello";**  **String s3=s1.intern();//returns string from pool, now it will be same as s2**  **System.out.println(s1==s2);//false because reference variables are pointing to different instance**  **System.out.println(s2==s3);//true because reference variables are pointing to same instance**  **}}**  **Output:**  **false**  **true**   1. **Java String indexOf():**   **public class IndexOfExample{**  **public static void main(String args[]){**  **String s1="this is index of example";**  **//passing substring**  **int index1=s1.indexOf("is");//returns the index of is substring**  **int index2=s1.indexOf("index");//returns the index of index substring**  **System.out.println(index1+" "+index2);//2 8**    **//passing substring with from index**  **int index3=s1.indexOf("is",4);//returns the index of is substring after 4th index**  **System.out.println(index3);//5 i.e. the index of another is**    **//passing char value**  **int index4=s1.indexOf('s');//returns the index of s char value**  **System.out.println(index4);//3**  **}}**  **Output:**  **2 8**  **5**  **3**   1. **Java String indexOf(String substring):**   **public class IndexOfExample2 {**  **public static void main(String[] args) {**  **String s1 = "This is indexOf method";**  **// Passing Substring**  **int index = s1.indexOf("method"); //Returns the index of this substring**  **System.out.println("index of substring "+index);**  **}**    **}**  **Output:**  **index of substring 16**   1. **Java String indexOf(String substring, int fromIndex) Method Example**   **public class IndexOfExample3 {**  **public static void main(String[] args) {**  **String s1 = "This is indexOf method";**  **// Passing substring and index**  **int index = s1.indexOf("method", 10); //Returns the index of this substring**  **System.out.println("index of substring "+index);**  **index = s1.indexOf("method", 20); // It returns -1 if substring does not found**  **System.out.println("index of substring "+index);**  **}**  **}**  **Output:**  **index of substring 16**  **index of substring -1**   1. **Java String indexOf(int char, int fromIndex) Method Example**   **public class IndexOfExample4 {**  **public static void main(String[] args) {**  **String s1 = "This is indexOf method";**  **// Passing char and index from**  **int index = s1.indexOf('e', 12); //Returns the index of this char**  **System.out.println("index of char "+index);**  **}**  **}**  **Output:**  **index of char 17**   1. **Java String toLowerCase()**   **public class StringLowerExample{**  **public static void main(String args[]){**  **String s1="JAVATPOINT HELLO stRIng";**  **String s1lower=s1.toLowerCase();**  **System.out.println(s1lower);**  **}}**  **Output:**  **javatpoint hello string**   1. **Java String toLowerCase(Locale locale)**   **import java.util.Locale;**  **public class StringLowerExample2 {**  **public static void main(String[] args) {**  **String s = "JAVATPOINT HELLO stRIng";**  **String eng = s.toLowerCase(Locale.ENGLISH);**  **System.out.println(eng);**  **String turkish = s.toLowerCase(Locale.forLanguageTag("tr")); // It shows i without dot**  **System.out.println(turkish);**  **}**  **}**  **Output:**  **javatpoint hello string**  **javatpo?nt hello str?ng**   1. **Java String toUpperCase()**   **public class StringUpperExample{**  **public static void main(String args[]){**  **String s1="hello string";**  **String s1upper=s1.toUpperCase();**  **System.out.println(s1upper);**  **}}**  **Test it Now**  **Output:**  **HELLO STRING**   1. **Java String toUpperCase(Locale locale)**   **import java.util.Locale;**  **public class StringUpperExample2 {**  **public static void main(String[] args) {**  **String s = "hello string";**  **String turkish = s.toUpperCase(Locale.forLanguageTag("tr"));**  **String english = s.toUpperCase(Locale.forLanguageTag("en"));**  **System.out.println(turkish);//will print I with dot on upper side**  **System.out.println(english);**  **}**  **}**  **Output:**  **HELLO STR?NG**  **HELLO STRING**   1. **Java String Trim() :**   **public class StringTrimExample{**  **public static void main(String args[]){**  **String s1=" hello ";**  **System.out.println(s1+"how are you"); // without trim()**  **System.out.println(s1.trim()+"how are you"); // with trim()**  **}}**  **In the above code, the first print statement will print “hello how are you” while the second statement will print “hellohow are you” using the trim() function.** |

1. **Java String ValueOf():**

**public class StringValueOfExample{**

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

**int value=20;**

**String s1=String.valueOf(value);**

**System.out.println(s1+17); //concatenating string with 10**

**}}**

**In the above code, it concatenates the Java String and gives the output – 2017.**