CORE JAVA

MANUAL V8.3

MODULE CODE:

ANUDIP FOUNDATION

ICONS AND THEIR MEANING
HINTS:
Get ready for helpful insites on difficult topics and questions.
STUDENTS:
This icon symbolize important instrcutions and guides for the students.
TEACHERS/TRAINERS:
This icon symbolize important instrcutions and guides for the trainers.

Module 3: String Handling, Regular Expression and Wrapper class

Chapter 2

Objective: After completing this lesson you will be able to : * Learn about Java predefined functions. * Gain an understanding of the StringTokenizer Class in Java	**Materials Required:** 1. Computer 2. Internet access
Theory Duration: 60 minutes	**Practical Duration:** 60 minutes
Total Duration: 120 minutes	

Chapter 2

2.1 Predefined String Functions

The String class in Java has a set of predefined functions for performing specific operations. Take a look at the functions mentioned below and learn about their uses.

Predefined String Functions in Java

* **length**()

The length() function of a string class returns the length of the string.

```
String str='No';
System.out.println(str.length());
Output : 2
```

* **charAt**()

The charAt() string class function returns characters located at a particular index location.

```
String str = 'Java';
System.out.println(str.charAt(2));
Output : v
```

* **equalsIgnoreCase**()

equalsIgnoreCase() performs a comparison of two strings, It ignores case differences.

```
String str = 'Guava';
System.out.println(str.equalsIgnoreCase('GUAVA'));
```

Output : true

* **toLowerCase**()

This method returns the string in lowercase. It converts all string characters into lower case letters.

```
String str = ‘Hello‘;
System.out.println(str.toLowerCase());
Output : hello
```

* **trim**()

The method returns a copy of the string while omitting trailing whitespace and leading.

```
String str = ‘   Hello  ‘;
System.out.println(str.trim());
Output : Hello
```

* **toUpperCase**()

Converts all string characters to uppercase using default locale rules.

```
String str = ‘Hello‘;
System.out.println(str.toUpperCase());
Output : HELLO
```

* **replace**()

This method replaces character occurrences with a particular new character.

```
String str = ‘Hello‘;
System.out.println(str.replace(’o’,’O));
```

Output : Hello

* **indexOf(String str)**

Returns the index of the first specified substring appearance within a string.

```
String str = ‘Java‘;
System.out.println(str.indexOf(‘v‘));
Output : 2
```

* **split(String regex)**

The split method splits a string based on the provided expression. It returns a string array.

```
String str = ‘Java‘;
String[] strArray = str.split(‘v‘);
for(String eachString:strArray){
        System.out.println(eachString);
}
Output : Ja
a
```

*** startsWith(String prefix)**

The method tests whether a string starts with a particular prefix. If the string character chain of the argument is a prefix of the string character chain, a true value is the output. If not, the value is false.

```
String str = ‘Hello‘;
System.out.println(str.startsWith(‘ja‘));
Output : false
```

* **substring()**

The substring() method returns a section of the string. Two substring methods are –

```
public String substring(int beginIndex);

public String substring(int beginIndex, int endIndex);
```

2.2 String Tokenizer class

The Java StringTokenizer class (**java.util.StringTokenizer**) lets programmers break down a string into tokens.

Java String Tokenizer example

```
import java.util.StringTokenizer;
        public class Simple{
         public static void main(String args[]){
           StringTokenizer st = new StringTokenizer('I am a man',' ');
             while (st.hasMoreTokens()) {
                 System.out.println(st.nextToken());
             }
           }
        }
```

Output:
```
I
am
a
man
```

Java String TokenizerConstructor

* **StringTokenizer(String str)** – creates a StringTokenizer containing a particular string.

* **StringTokenizer(String delim, String str)** – creates a StringTokenizer with a particular string and delimiter.

* **StringTokenizer(String delim, String str, boolean returnValue) – creates a** StringTokenizer with a designated delimiter, string and returnValue. If the return value is true, the characters of delimiter are tokens. Otherwise, the same characters cater to different tokens.

StringTokenizer Methods

boolean hasMoreTokens() – checks for the availability of more tokens.

String nextToken() – returns the next StringTokenizer object.

String nextToken(String delim) – returns the next delimiter-based token.

boolean hasMoreElements() – identical to the hasMoreTokens() method.

Object nextElement() – similar to nextToken() but returns object.

int countTokens() – returns value displaying the total count of tokens.

Practical (60 minutes)

See the example programme for Java StringTokenizer class below. Write the same programme for the class StringT, to output the tokenized value for the string "You are a woman". Write the same programme for the class Stringy, to output the value for the string "Tomorrow is Monday".

```
import java.util.StringTokenizer;

          public class Simple{

           public static void main(String args[]){

             StringTokenizer st = new StringTokenizer(‘I am a man‘,‘ ‘);

               while (st.hasMoreTokens()) {

                   System.out.println(st.nextToken());

               }

             }

}
```

Instructions: The progress of students will be assessed with the exercises mentioned below.

MCQ

1. Predefined String class functions perform specific _________.

a) exclusions

b) operations

c) denominations

d) None of the mentioned

2. The length() function of a string class returns the __________.

a) breadth

b) length

c) dimensions

d) None of the mentioned

3. The charAt() function returns characters located at a particular ______ location.

a) nested

b) standalone

c) index

d) None of the mentioned

4. The equalsIgnoreCase() performs the _________ of two strings.

a) comparison

b) examination

c) diversification

d) combination

5. Which function converts string characters into lower case letters?

a) indexOf(String str)

b) toLowerCase()

c) startsWith(String prefix)

d) None of the mentioned

6. Which function can be used to return a string copy without whitespace?

a) trim()

b) equalsIgnoreCase()

c) replace()

d) split(String regex)

7. The replace() is used to replace __________.

a) threads

b) characters

c) classes

d) None of the mentioned

8. What does the split(String regex) function split?

a) strings

b) threads

c) classes

d) subclasses

9. The startsWith(String prefix) checks if a string begins with a specific _________.

a) suffix

b) prefix

c) digit

d) None of the mentioned

10. java.util.StringTokenizer lets programmers break down a string into __________.

a) Substrings

b) sub-tokens

c) tokens

d) None of the mentioned