Course Name - Object Oriented Programming using Java

Lecture 16 – Methods of String Class (Contd.) and Methods of StringBuffer Class

Presented By
Dr. Sudipta Sahana
Asso. Prof.
Dept. of CSE
UEM - Kolkata

Topic of Interest

- **toCharArray()**
- **toString()**
- **toLowerCase()**
- **toUpperCase()**
- **trim()**
- **valueOf()**
- **append()**
- **delete()**
- **deleteCharAt()**
- **capacity()**
- **charAt()**

toCharArray()

This method converts the given string into a sequence of characters. The returned array length is equal to the length of the string. It returns a newly allocated character array.

```
class Gfg {
    public static void main(String
args[])
    {
        String s = "HelloJISCE";
        char[] gfg = s.toCharArray();
        for (int i = 0; i < gfg.length;
i++) {
            System.out.println(gfg[i]);
        }
    }
}
```

Output

```
H
e
l
l
o
J
I
S
C
E
```

toString()

If you want to represent any object as a string, toString() method comes into existence. The toString() method returns the string representation of the object. If you print any object, java compiler internally invokes the toString() method on the object. So overriding the toString() method, returns the desired output, it can be the state of an object etc. depends on your implementation.

```
class Student{
 int rollno;
 String name;
 String city;
 Student(int rollno, String name, String
city){
 this.rollno=rollno;
 this.name=name;
 this.city=city;
 }
 public String toString(){//overriding
//the toString() method
return rollno+" "+name+" "+city;
 }
 public static void main(String args[]){
   Student s1=new Student(101,"Raj","lucknow");
   Student s2=new
Student(102,"Vijay","ghaziabad");
   System.out.println(s1);//compiler writes here
s1.toString()
   System.out.println(s2);//compiler writes here
s2.toString()
 } }
```

Output

```
101 Raj lucknow
102 Vijay Ghaziabad
```

toLowerCase()

This method is used to convert upper case string into lower case.

```
class StringHandling
{
public static void main(String arg[])
{
String s="JAVA";
System.out.println("String:
"+s.toLowerCase());
}
}
```

Output

String: java

toUpperCase()

This method is used to convert lower case string into upper case..

```
public class StringUpperExample{
public static void main(String
args[]){
String s1="hello string";
String s1upper=s1.toUpperCase();
System.out.println(s1upper);
}
}
```

Output

HELLO STRING

trim()

The java string trim() method eliminates leading and trailing spaces. The unicode value of space character is '\u0020'. The trim() method in java string checks this unicode value before and after the string, if it exists then removes the spaces and returns the omitted string.

```
public class StringTrimExample{
public static void main(String args[]){
String s1="  hello string   ";
System.out.println(s1+"Tiger");//without trim()
System.out.println(s1.trim()+"Tiger");//with trim()
}}
```

Output

```
   hello string   Tiger
hello stringTiger
```

valueOf()

This method converts different types of values into string. By the help of string valueOf() method, you can convert int to string, long to string, boolean to string, character to string, float to string, double to string, object to string and char array to string.

```
 public class StringValueOfExample{
public static void main(String args[]){
int value=30;
String s1=String.valueOf(value);
System.out.println(s1+10);//concatenati
ng string with 10
}}
```

Output

3010

append()

This method is used to add a new string at the end of original string.

```
class StringHandling
{
public static void main(String arg[])
{
StringBuffer sb=new StringBuffer("java is
easy");
System.out.println(sb.append(" to learn"));
}
}
```

Output

java is easy to learn

delete()

This method is used to delete a string from the given string based on index value.

```
class StringHandling
{
public static void main(String arg[])
{
StringBuffer sb=new StringBuffer("java is
easy to learn");
StringBuffer s;
s=sb.delete(8, 13);
System.out.println(s);
}
}
```

Output

java is to learn

deleteCharAt()

This method is used to delete a character at given index value

```
class StringHandling
{
public static void main(String arg[])
{
StringBuffer sb=new StringBuffer("java");
System.out.println(sb.deleteCharAt(3));
}
}
```

Output

jav

capacity()

This method returns the current capacity. The capacity is the amount of storage available for newly inserted characters, beyond which an allocation will occur.

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

      StringBuffer buff = new StringBuffer("ObjectOriented");
// returns the current capacity of the String buffer i.e. 16 + 14
      System.out.println("capacity = " + buff.capacity());
 buff = new StringBuffer(" ");
// returns the current capacity of the String buffer i.e. 16 + 1
       System.out.println("capacity = " + buff.capacity());
   }
}
```

Output

```
capacity = 30
capacity = 17
```

charAt()

This method returns the char value in this sequence at the specified index. The first char value is at index 0, the next at index 1, and so on, as in array indexing.
The index argument must be greater than or equal to 0, and less than the length of this sequence.

```
public class StringBufferDemo {
   public static void main(String[] args) {
      StringBuffer buff = new StringBuffer("Object Oriented");
      System.out.println("buffer = " + buff);
 System.out.println("character = " + buff.charAt(4));
   buff = new StringBuffer("amrood admin ");
      System.out.println("buffer = " + buff)
      System.out.println("character = " + buff.charAt(6));
   }
}
```

Output

```
buffer = Object Oriented
character = c
buffer = amrood admin
character =
```

Thank You