String, StringBuffer And StringBuilder

Q. What is the String in java?

String is immutable class of java. In java every “ ” consider as string object

What is the immutable class?

Immutable means once we assign value we cannot change later called as immutable.

If we want to work with string in java we have the two ways

a) By using initialization technique:

syntax: String variablename=”values”;

e.g String s=”Good”;

b) By using new keyword :

syntax: String variablename = new String(“value”);

e.g String str = new String(“Good”);

Q. what is the diff between above mention approaches?

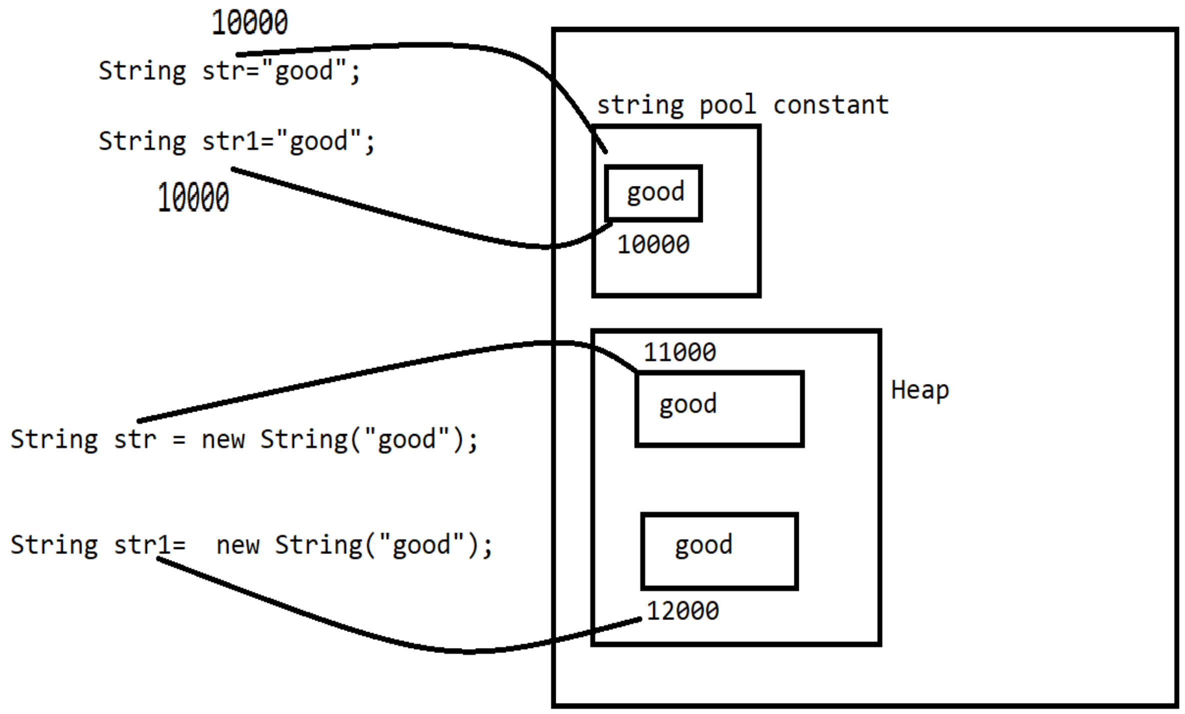
If we use the initialization technique for string then string object get created in string pool constant and if we use the new keyword for string object creation then string object get created in heap section.

What is the string pool constant?

String pool constant is part of memory in heap which specially design for store the object initialized by string the benefit is if we have two strings of same value and if we initialize it then JVM not create the separate memory for different object allocate single memory for all objects and use the same reference or address in all different variables.

But if use the new keyword and if we multiple objects with same value then JVM create the new object every time.

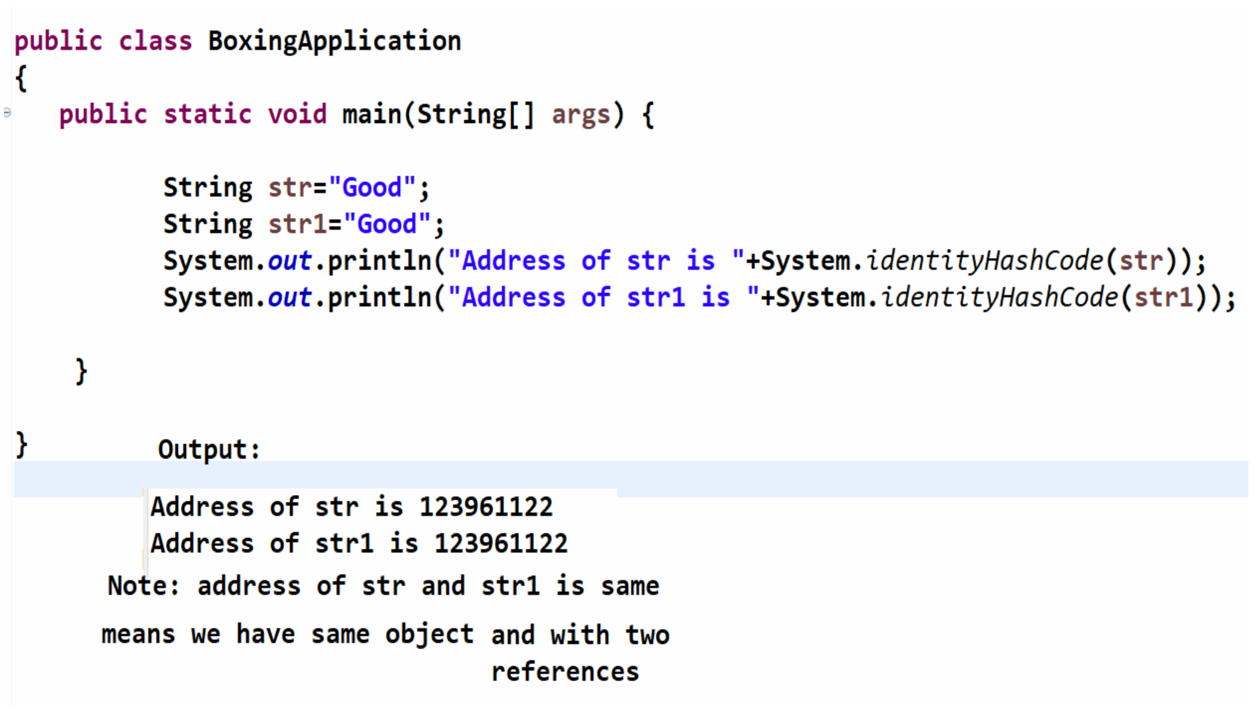
Following Diagram show the meaning of above statements



As per above diagram if we think about string constant diagram we have the two string name as str and str1 with value "good" so JVM create the single object of both str and str1 and share the address of "good" object to str and str1.

If we think about the heap space diagram we have the strings name as str and str1 with "good" value and JVM create two different objects of same string.

String with Initialization approach



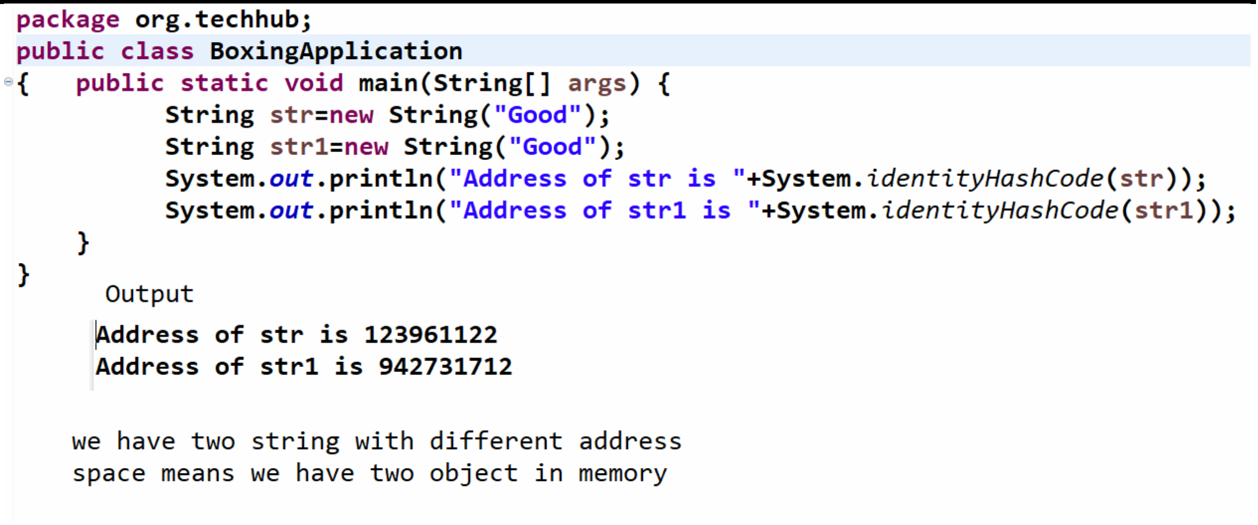
```java
public class BoxingApplication
{
    public static void main(String[] args) {

            String str="Good";
            String str1="Good";
            System.out.println("Address of str is "+System.identityHashCode(str));
            System.out.println("Address of str1 is "+System.identityHashCode(str1));

    }

}
```

Output:

```
Address of str is 123961122
Address of str1 is 123961122
```

Note: address of str and str1 is same means we have same object and with two references

String with new keyword approach



```java
package org.techhub;
public class BoxingApplication
{   public static void main(String[] args) {
            String str=new String("Good");
            String str1=new String("Good");
            System.out.println("Address of str is "+System.identityHashCode(str));
            System.out.println("Address of str1 is "+System.identityHashCode(str1));
    }
}
```

Output

```
Address of str is 123961122
Address of str1 is 942731712
```

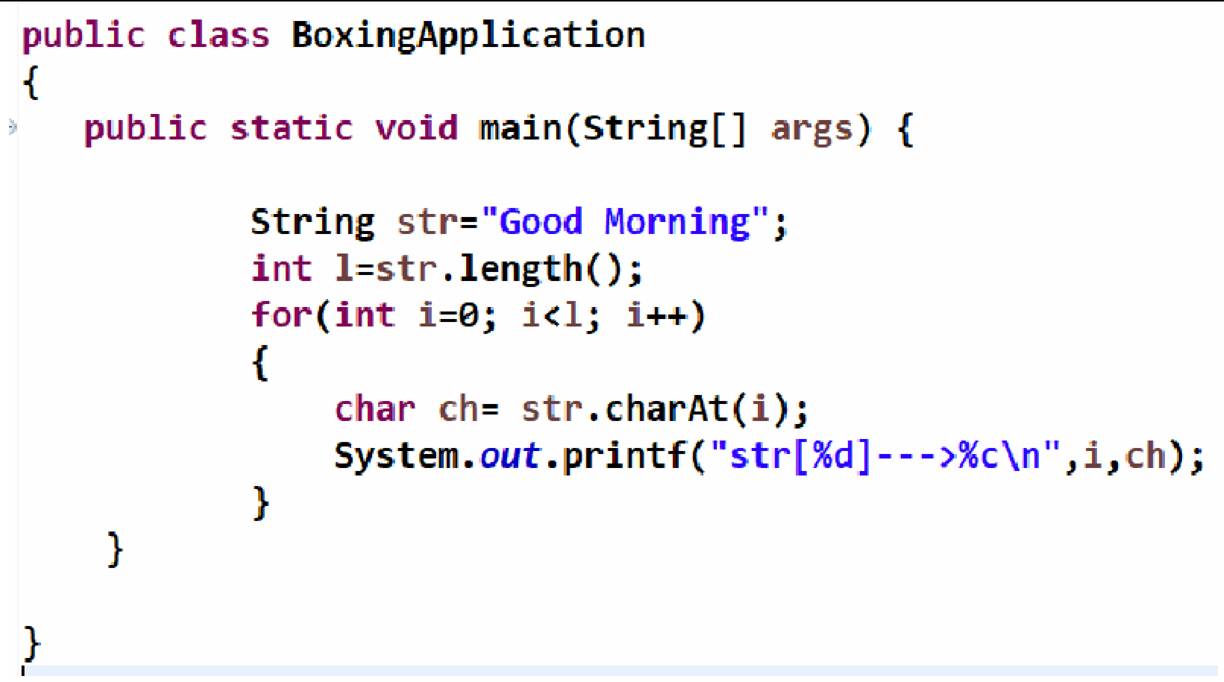
we have two string with different address space means we have two object in memory

String class methods

String class provide the some inbuilt method to us for work with string

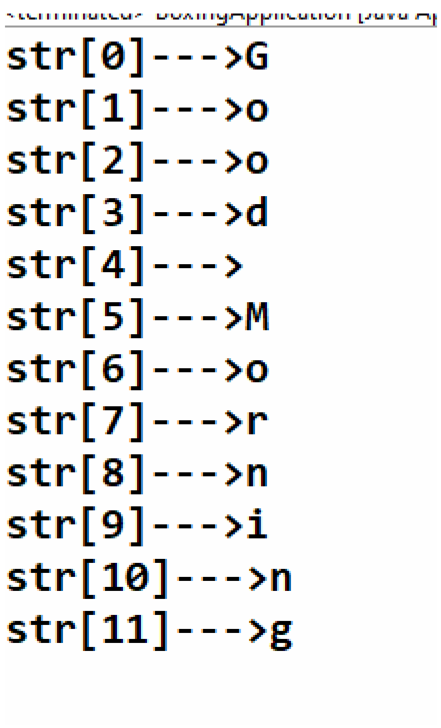
int length(): this is used for calculate the length of string
char charAt(int index): this is used for return character from its index.

Example



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

            String str="Good Morning";
            int l=str.length();
            for(int i=0; i<l; i++)
            {
                char ch= str.charAt(i);
                System.out.printf("str[%d]--->%c\n",i,ch);
            }
    }

}
```

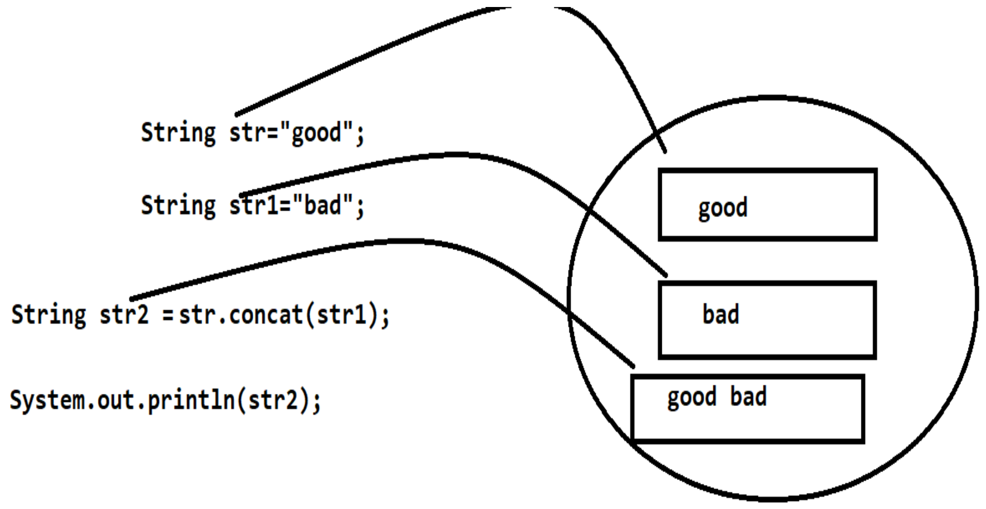
Output



```
str[0]--->G
str[1]--->o
str[2]--->o
str[3]--->d
str[4]--->
str[5]--->M
str[6]--->o
str[7]--->r
str[8]--->n
str[9]--->i
str[10]--->n
str[11]--->g
```

strcat (): this is used for concat the two string with each other and generate the third new string from it.

Following Diagram shows the working strcat() method

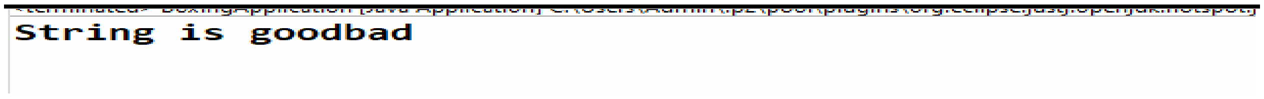


```
String str="good";

String str1="bad";

String str2 =str.concat(str1);

System.out.println(str2);
```

Example

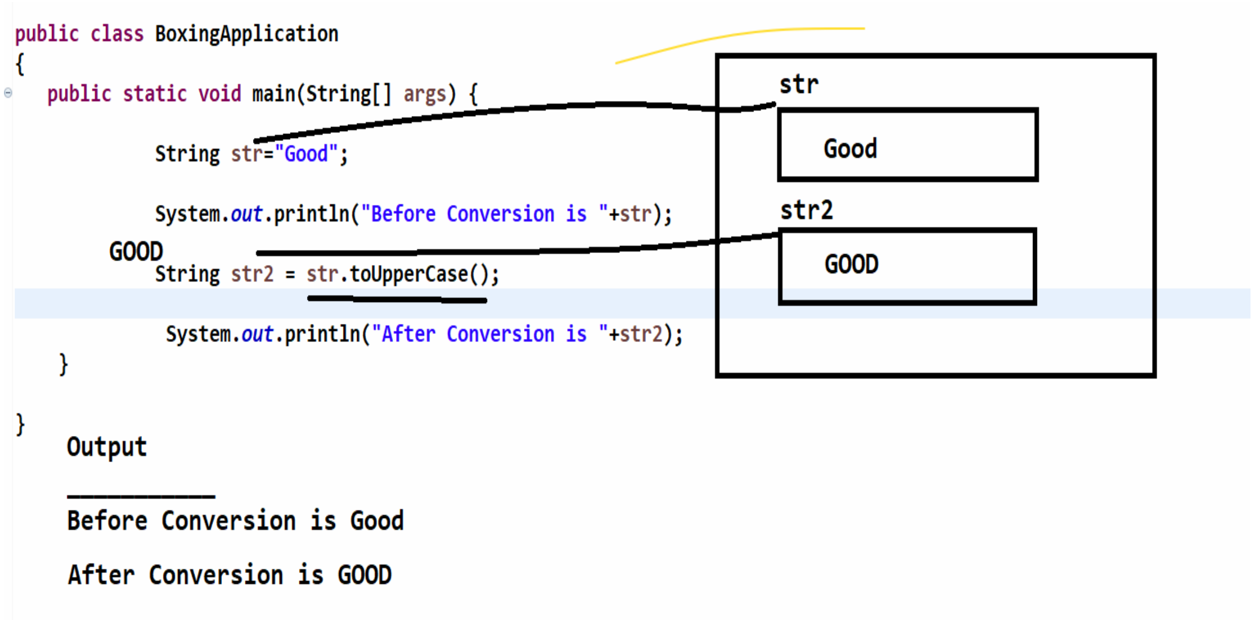
```
public class BoxingApplication
{
  public static void main(String[] args) {

                String str="good";
                String str1="bad";

               String str2=str.concat(str1);
                System.out.println("String is "+str2);
       }

}
```

Output



```
String is goodbad
```

String toUpperCase(): this is used for convert the lower case string to upper case string.



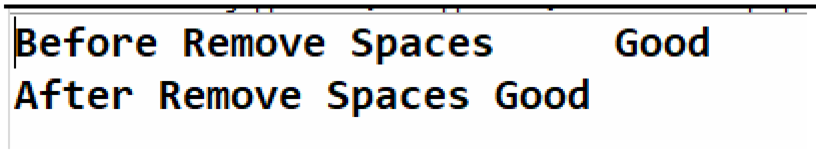
```
public class BoxingApplication
{
   public static void main(String[] args) {

          String str="Good";

          System.out.println("Before Conversion is "+str);

          String str2 = str.toUpperCase();

           System.out.println("After Conversion is "+str2);
   }

}
```

```
Output
__________
Before Conversion is Good

After Conversion is GOOD
```

String trim (): this method is used for remove the white spaces from string at beginning and ending.

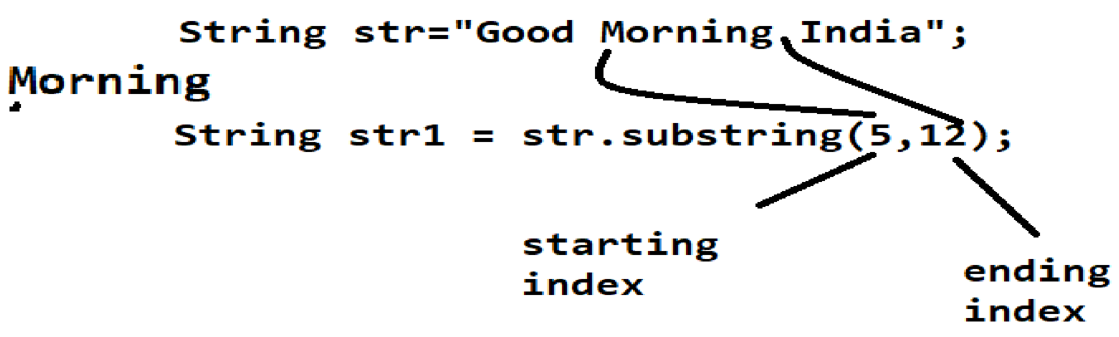
```
public class BoxingApplication
{   public static void main(String[] args) {
          String str= "    Good";
          System.out.println("Before Remove Spaces "+str);
          String str1=str.trim();
          System.out.println("After Remove Spaces "+str1);
      }

}
```

Output

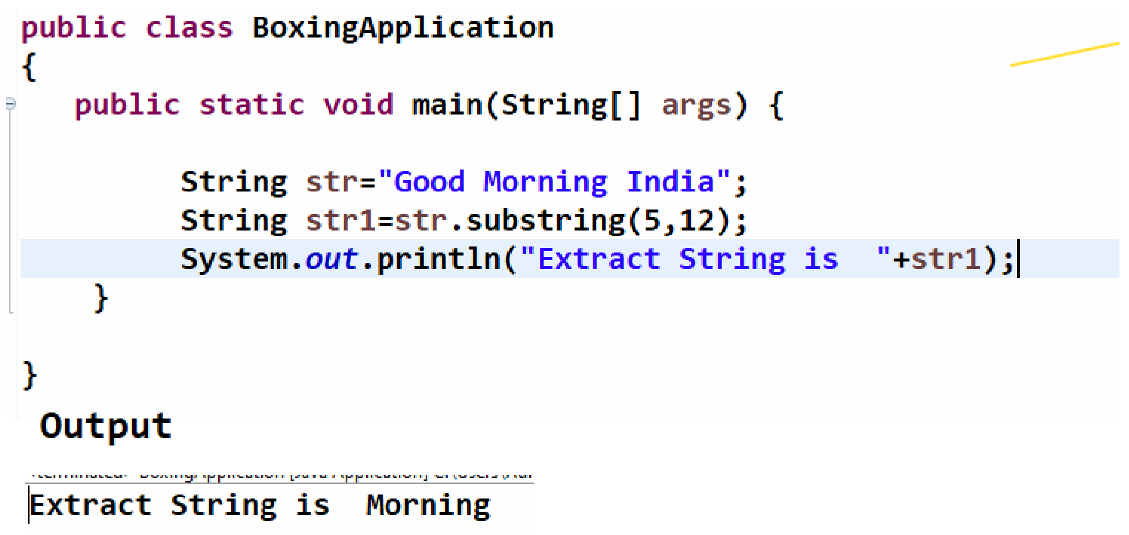


```
Before Remove Spaces     Good
After Remove Spaces Good
```

String substring(int start,int end): this method is used for extract the some specified portion from string.


String str="Good Morning India";
Morning
String str1 = str.substring(5,12);
starting index
ending index

Sample code



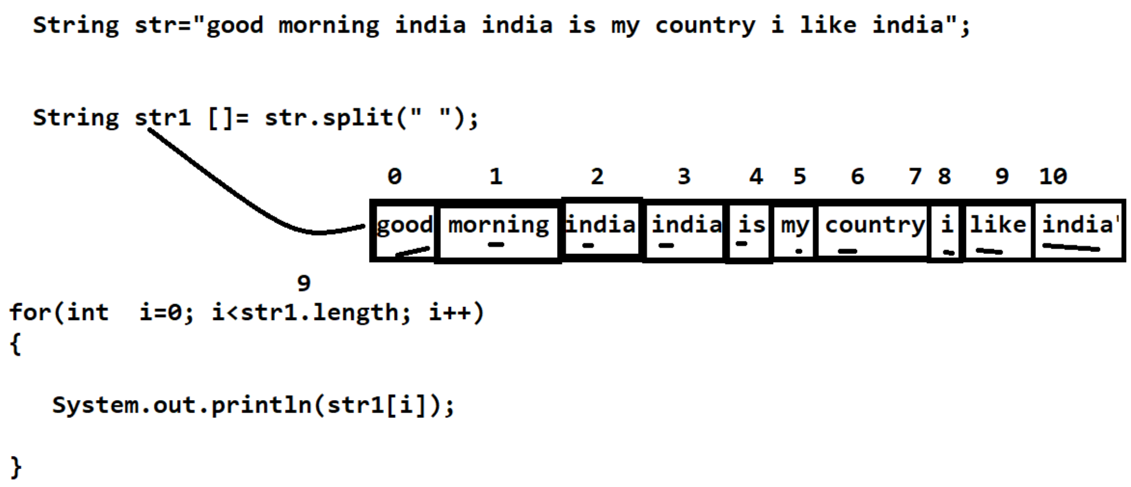
```
public class BoxingApplication
{
   public static void main(String[] args) {

          String str="Good Morning India";
          String str1=str.substring(5,12);
          System.out.println("Extract String is  "+str1);
    }

}
```

Output

```
Extract String is  Morning
```

String [] split(String character): this method is used split the string using some specified character.

Following Diagram shows the working split method

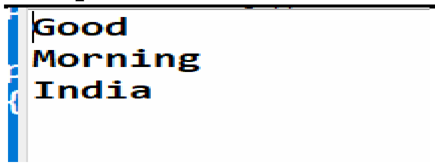


```
String str="good morning india india is my country i like india";

String str1 []= str.split(" ");

for(int  i=0; i<str1.length; i++)
{

   System.out.println(str1[i]);

}
```

Example

```
package org.techhub;

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

           String str="Good Morning India";
           String str1[]=str.split(" ");

           for(int i=0; i<str1.length;i++)
           {
             System.out.println(str1[i]);
           }
       }
}
```

Output



```
Good
Morning
India
```

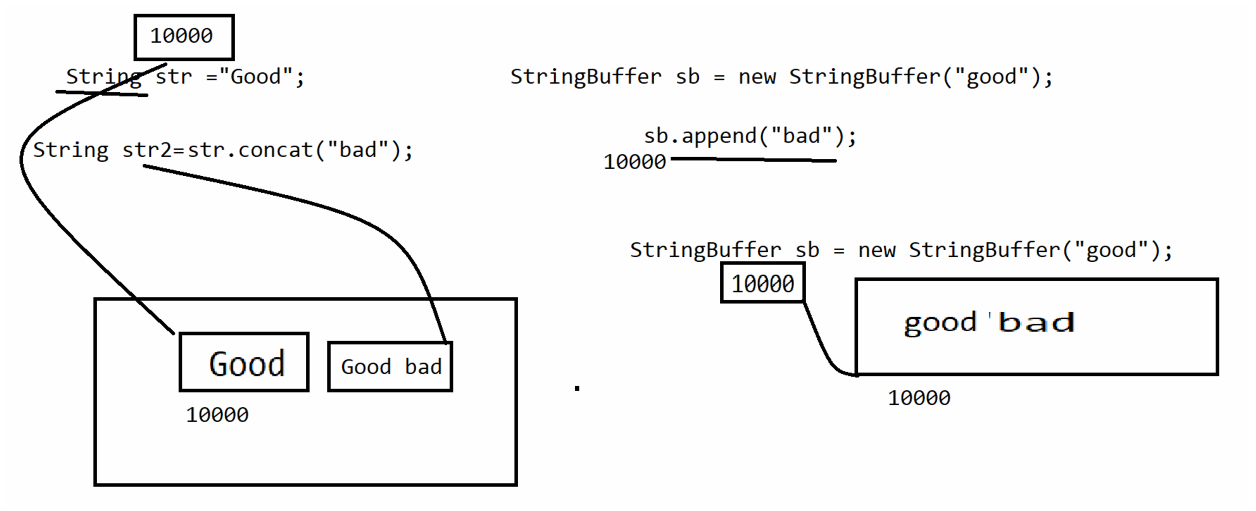
StringBuffer and StringBuilder

StringBuffer and StringBuilder are the mutable classes in java
Mutable means once we initialize value in it we can modify it called as mutable.
Note: we can use the StringBuffer and StringBuilder by using new keyword only**.**

StringBuffer and StringBuilder contain the some additional methods as per compare with string.

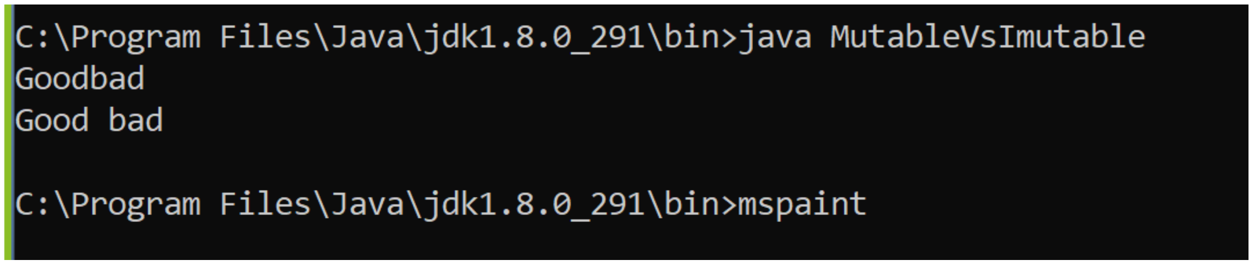
void insert(int index,int data): this method can append data on specified index in String
void insert(int index,float data): this is used for the floating data on specified
Note: this is the overloaded method with all data types for inserting data.
void append(int data): this method can add the new data at the end of string of type integer
void append(float data): this method can add the new data at the end of string of type float
Note: this is also overloaded method with all data types
void delete(int startindex,int endidex): this is used for delete the data between two specified index

Following Diagram shows the working of mutable and immutable


10000
String str ="Good";
String str2=str.concat("bad");
Good
Good bad
10000
StringBuffer sb = new StringBuffer("good");
sb.append("bad");
10000
StringBuffer sb = new StringBuffer("good");
10000
good bad
10000

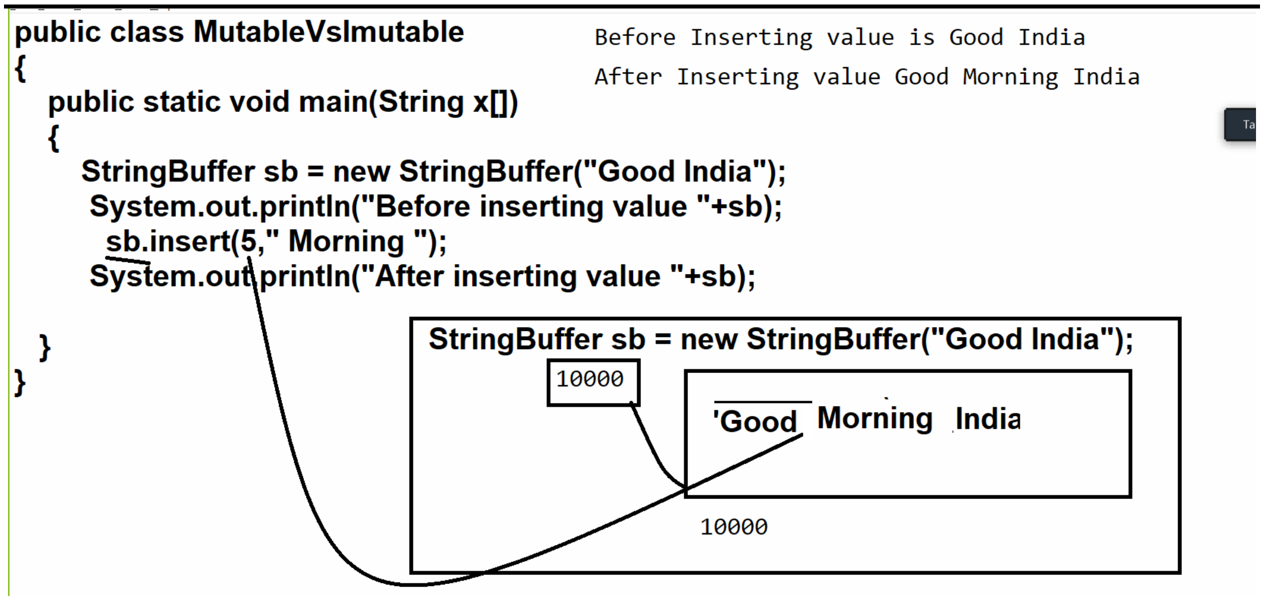
Source Code Example

```
public class MutableVsImutable
{
   public static void main (String x[])
   {   String str="Good";
      String str2=str.concat ("bad");
  System.out.println (str2);
   StringBuffer sb = new StringBuffer ("Good");
      sb.append (" bad");
  System.out.println (sb);
  }
}
```



```
C:\Program Files\Java\jdk1.8.0_291\bin>java MutableVsImutable
Goodbad
Good bad

C:\Program Files\Java\jdk1.8.0_291\bin>mspaint
```

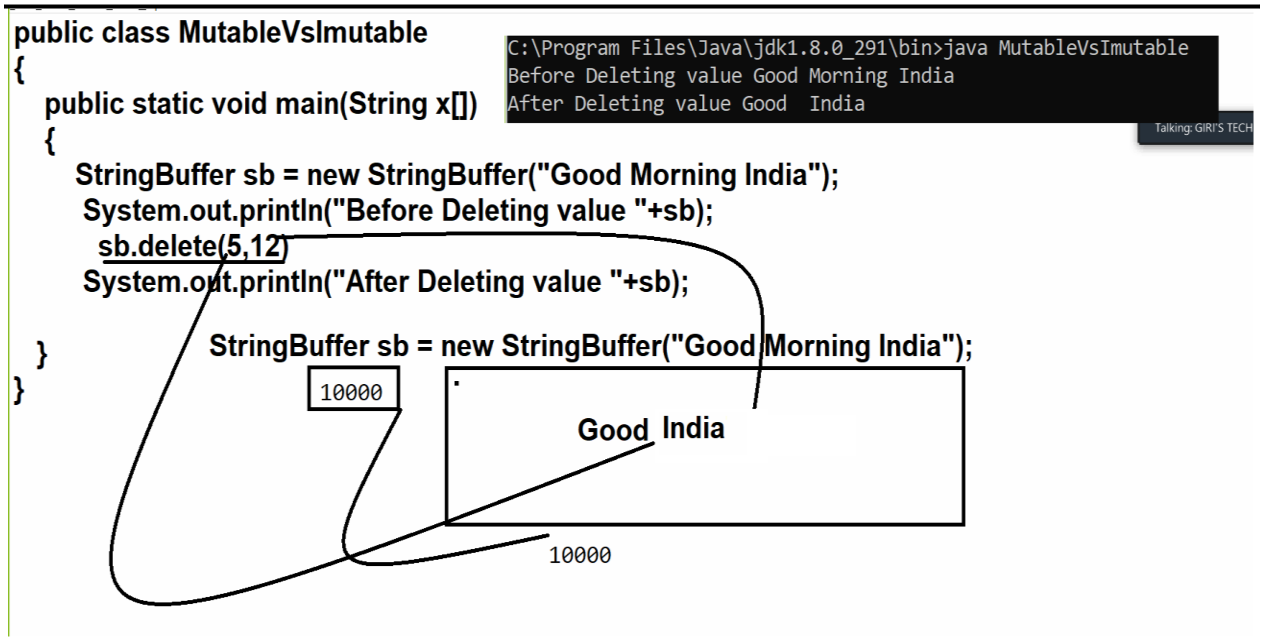
How to insert the value on specified index using StringBuffer



```
public class MutableVsImutable
{
   public static void main(String x[])
   {
      StringBuffer sb = new StringBuffer("Good India");
       System.out.println("Before inserting value "+sb);
         sb.insert(5," Morning ");
       System.out.println("After inserting value "+sb);

   }
}
```

```
Before Inserting value is Good India
After Inserting value Good Morning India
```

How to delete data between two specified indexes from StringBuffer



```
public class MutableVsImutable
{
   public static void main(String x[])
   {
     StringBuffer sb = new StringBuffer("Good Morning India");
      System.out.println("Before Deleting value "+sb);
        sb.delete(5,12)
      System.out.println("After Deleting value "+sb);

   }
}
```

```
C:\Program Files\Java\jdk1.8.0_291\bin>java MutableVsImutable
Before Deleting value Good Morning India
After Deleting value Good  India
```

What is the diff between StringBuffer and StringBuilder

The major diff between StringBuffer and StringBuilder is StringBuffer is synchronized and StringBuilder is not means StringBuffer is thread safe class and StringBuilder is not thread safe.