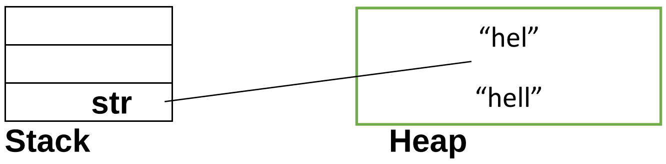
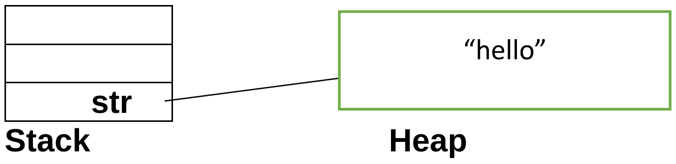
Java - Introduction to Programming
Lecture 8

ALWAYS REMEMBER: Java Strings are Immutable.

String str = “h”
Str + “e”
Str + “l”
Str+ “l”
Str + “o”



String Builder



Declaration

```
public class Stringbuilders {
public static void main(String[] args) {
    StringBuilder sb = new StringBuilder();
      Sb = "Piyush Keshari";
    System.out.println(sb);
}
}
```

Get A Character from Index

```
public class Stringbuilders {
public static void main(String[] args) {
    StringBuilder sb = new StringBuilder("Piyush Keshari");
    //Get Char
    System.out.println(sb.charAt(0));
}
}
```

Set a Character at Index

```java
public class Stringbuilders {
public static void main(String[] args) {
    StringBuilder sb = new StringBuilder("Piyush Keshari");
    //Set Char
    sb.setCharAt(0,'A');
    System.out.println(sb);
}
}
```

Insert a Character at Some Index

```java
public class Stringbuilders {
public static void main(String[] args) {
    StringBuilder sb = new StringBuilder("Piyush Keshari");
    //Insert Char
    sb.insert(4,'s');
    System.out.println(sb);
}
}
```

Delete char at some Index

```java
public class Stringbuilders {
public static void main(String[] args) {
    StringBuilder sb = new StringBuilder("Piyush Keshari");
    //delete Char
    sb.delete(4,5);
    System.out.println(sb);
}
}
```

Append a char

Append means to add something at the end.

```java
public class Stringbuilders {
public static void main(String[] args) {
    StringBuilder sb = new StringBuilder("h");
    sb.append("e");
    sb.append("l");
    sb.append("l");
    sb.append("o");
    System.out.println(sb);
}
}
```

Print Length of Stringbuilder

```java
public class Stringbuilders {
public static void main(String[] args) {
    StringBuilder sb = new StringBuilder("h");
    sb.append("e");
    sb.append("l");
    sb.append("l");
    sb.append("o");
    System.out.println(sb.length());
}
}
```

Difference Between String, StringBuilder, and StringBuffer

Feature	String	StringBuilder	StringBuffer
Introduction	Introduced in JDK 1.0	Introduced in JDK 1.5	Introduced in JDK 1.0
Mutability	Immutable	Mutable	Mutable
Thread Safety	Thread Safe	Not Thread Safe	Thread Safe
Memory Efficiency	High	Efficient	Less Efficient
Performance	High(No-Synchronization)	High(No-Synchronization)	Low(Due to Synchronization)
Usage	This is used when we want immutability.	This is used when Thread safety is not required.	This is used when Thread safety is required.

- Thread-safe means that multiple threads can access or modify something without causing errors, while non-thread-safe does not guarantee this.

```
// Java program to demonstrate difference between
// String, StringBuilder and StringBuffer
public class StrStrbuildStrBuff {

    // Method 1
    // Concatenates to String
    public static void concat1(String s1)
```

```java
    {
        s1 = s1 + "keshari";
    }

    // Method 2
    // Concatenates to StringBuilder
    public static void concat2(StringBuilder s2)
    {
        s2.append("keshari");
    }

    // Method 3
    // Concatenates to StringBuffer
    public static void concat3(StringBuffer s3)
    {
        s3.append("keshari");
    }

    // Method 4
    // Main driver method
    public static void main(String[] args)
    {
        // Custom input string
        // String 1
        String s1 = "piyush";

        // Calling above defined method
        concat1(s1);

        // s1 is not changed
        System.out.println("String: " + s1);

        // String 1
        StringBuilder s2 = new StringBuilder("piyush");

        // Calling above defined method
        concat2(s2);

        // s2 is changed
        System.out.println("StringBuilder: " + s2);

        // String 3
        StringBuffer s3 = new StringBuffer("piyush");

        // Calling above defined method
        concat3(s3);

        // s3 is changed
        System.out.println("StringBuffer: " + s3);
    }
}
```

Reverse a String (using reverse() with stringbuilder)

```
public class Stringbuilders {
public static void main(String[] args) {
    StringBuilder sb = new StringBuilder("piyush");
    sb.reverse();
    System.out.println(sb);
```

Reverse a String (using reverse() with stringbuffer)

```
public class Stringbuffers {
public static void main(String[] args) {
    StringBuffer sb = new StringBuffer("piyush");
    sb.reverse();
    System.out.println(sb);

}
}
```

Reverse a string without using reverse()

```
public class Strings {
public static void main(String[] args) {
    String str = new String("piyush");
    String revstr = "";
    for(int i=str.length()-1;i>=0;i--){
        revstr = revstr + str.charAt(i);
    }
    System.out.println(revstr);

}
}
```

Remove duplicate characters from the string

```
public class StrStrbuildStrBuff {
public static void main(String[] args) {
    String str = "programming";
    StringBuilder sb = new StringBuilder();
    for(int i=0;i<str.length();i++){
        char ch = str.charAt(i);
        int indx = str.indexOf(ch,i+1);
        if(indx==-1){
            sb.append(ch);
        }
    }
    System.out.println(sb);
    }
}
```

2nd Approach using arrays

```
public class StrStrbuildStrBuff {
public static void main(String[] args) {
    String str = "programming";
    char[] arr = str.toCharArray();
    StringBuilder sb = new StringBuilder();
    for(int i=0;i<arr.length;i++){
        boolean repeated = false;
        for(int j=i+1;j<arr.length;j++){
            if(arr[i]==arr[j]){
                repeated=true;
                break;
            }
        }
        if(!repeated){
            sb.append(arr[i]);
        }

    }
    System.out.println(sb);
    }
}
```

Reverse Each word of a string

```
public class StrStrbuildStrBuff {
public static void main(String[] args) {
    String str = "piyush keshari";
    String rstr = "";
    String[] words = str.split(" ");
    for(String word: words){
        String revword = "";
        for(int i = word.length()-1;i>=0;i--){
            revword = revword + word.charAt(i);
        }
        rstr = rstr+revword+" ";
    }
    System.out.println(rstr);
    }
}
```

Find Each Characater occurrence in String

```
import java.util.*;
```

```java
public class StrStrbuildStrBuff {
public static void main(String[] args) {
    String str = "piyushkeshari";
    Map<Character, Integer>map = new HashMap<>();
    char[] arr = str.toCharArray();
    for(char ch :arr){
        if(!map.containsKey(ch)){
            map.put(ch, 1);
        }
        else{
            int val = map.get(ch);
            map.put(ch, val+1);
        }
    }
    System.out.println(map);
    }
}
```

Reverse a String word by word

```java
import java.util.*;

class GfG {

    static String reverseWords(String str) {

        List<String> words = new ArrayList<>();
        String[] parts = str.split(" ");

        for (String word : parts) {
            if (!word.isEmpty()) {

                // Ignore empty words caused by multiple dots
                words.add(word);
            }
        }

        // Reverse the words
        Collections.reverse(words);

        // Join the reversed words back into a string
        return String.join(" ", words);
    }

    public static void main(String[] args) {
        String str = "My Name is piyush";
        System.out.println(reverseWords(str));
    }
}
```

Homework Problems

Try Solving all the String questions with StringBuilder.