Difference Between String, StringBuilder and StringBuffer Classes

String

String is *immutable* (once created can not be changed) object. The object created as a String is stored in the **Constant String Pool**. Every immutable object in Java is thread safe, that implies String is also thread safe. String can not be used by two threads simultaneously. String once assigned can not be changed.

StringBuffer

StringBuffer is mutable means one can change the value of the object. The object created through StringBuffer is stored in the heap. StringBuffer has the same methods as the StringBuilder, but each method in StringBuffer is synchronized that is StringBuffer is thread safe.

Due to this it does not allow two threads to simultaneously access the same method. Each method can be accessed by one thread at a time.

But being thread safe has disadvantages too as the performance of the StringBuffer hits due to thread safe property. Thus StringBuilder is faster than the StringBuffer when calling the same methods of each class. StringBuffer can be converted to the string by using toString() method.

StringBuilder demo1 = new StringBuffer("Hello");
// The above object stored in heap and its value can be changed.
demo1=new StringBuffer("Bye");
// Above statement is right as it modifies the value which is allowed in the StringBuffer

StringBuilder

StringBuilder is same as the StringBuffer, that is it stores the object in heap and it can also be modified. The main difference between the StringBuffer and StringBuilder is that **StringBuilder is also not thread safe**.

StringBuilder is fast as it is not thread safe.

StringBuilder demo2= new StringBuilder("Hello");
// The above object too is stored in the heap and its value can be modified
demo2=new StringBuilder("Bye");
// Above statement is right as it modifies the value which is allowed in the StringBuilder