StringBuilder and StringBuffer

StringBuilder: StringBuilder is a class in Java that represents a mutable sequence of characters. Unlike the String class, which creates immutable sequences, StringBuilder allows modification of its contents without creating new objects. It is specifically designed for use in single-threaded environments, as it does not provide thread safety. StringBuilder is a faster alternative to StringBuffer when thread synchronization is not required. Its functions are very similar to those of the StringBuffer class, making it a straightforward option for mutable string manipulation.

- Use StringBuilder in single-threaded environments for faster performance.
- Ideal for applications where thread safety is not a concern.

StringBuffer: StringBuffer is a class in Java that also represents a mutable sequence of characters. However, unlike StringBuilder, it provides thread safety by ensuring synchronization. This makes StringBuffer a suitable choice for multi-threaded environments where multiple threads may access or modify the string concurrently. Although it has slightly more overhead than StringBuilder due to synchronization, it ensures reliable operation in concurrent scenarios. StringBuffer is often used when both mutability and thread safety are essential.

Difference Between StringBuilder and StringBuffer

Feature	**StringBuilder**	**StringBuffer**
Thread Safety	Not thread-safe.	Thread-safe.

Synchronization	Does not provide synchronization.	Provides synchronization.
Performance	Faster, as it doesn't have overhead for thread safety.	Slower due to synchronization overhead.
Use Case	Single-threaded environment.	Multi-threaded environment.
Recommendation	Preferred for single-threaded programs.	Used when thread safety is required.

Here is the Example for the StringBuilder and StringBuffer Classes:

```java
// Java code to illustrate the internal
// working of String, StringBuilder
// and StringBuffer class
class GfG
{
    public static void main (String[] args) {
        String s1 = "geeks";

        String s2 = s1;

        // Creates a new location to store s1
        s1 = s1 + "forgeeks";

        // s1 and s2 refers to different location
        if(s1 == s2)
            System.out.println("Same");
        else
            System.out.println("Not Same");

        // StringBuilder or StringBuffer class
        StringBuilder sb1 = new StringBuilder("geeks");

        // sb2 refers to the same location as sb1
        StringBuilder sb2 = sb1;

        // Append operation modifies the same object
        // as it is mutable in nature
        sb1 = sb1.append("forgeeks");

        // Both sb1 and sb2 refers to the same location
        if(sb1 == sb2)
            System.out.println("Same");
        else
            System.out.println("Not Same");
    }
}
```

Output

- Not Same

Same