**String vs String builder and String buffer:**

In Java, Strings are sequences of characters stored in the heap memory's String pool, a specialized area dedicated to holding unique String literal values. Strings are immutable, meaning their values cannot change after creation. This immutability ensures thread safety and allows strings to be safely shared among multiple threads without risk of modification. It also enhances performance by allowing strings to be cached and reused efficiently from String pool, contributing to a more efficient use of memory.

StringBuilder and StringBuffer are indeed mutable, which means their content can be changed after creation. However, unlike immutable String objects, StringBuilder and StringBuffer instances are not stored in the string pool. Instead, they reside in the heap memory. This mutability allows efficient modification of their content, making them ideal for scenarios requiring dynamic string manipulation. While String objects are immutable for thread safety and efficient memory management, StringBuilder and StringBuffer sacrifice immutability for flexibility, enabling developers to directly modify string content.

**String builder vs String buffer:**

String-builder is not thread-safe but provides better performance in single-threaded environments.

String buffer used in multi-threaded environments where thread safety is required.but may incur some performance overhead due to synchronization.