String Manipulation vs StringBuilder in Java

1. String Manipulation in C#

In C#, strings are immutable objects of the `System.String` class. Modifying a string creates a new string in memory.

Common operations include:

```
// Concatenation
string s1 = "Hello";
string s2 = "World";
string result = s1 + " " + s2;

// Substring
string sub = result.Substring(0, 5); // "Hello"

// Replace
string replaced = result.Replace("World", "C#");

// Length
int len = result.Length;

// Case Conversion
string upper = result.ToUpper();
string lower = result.ToLower();

// Trimming whitespace
string trimmed = result.Trim();
```

2. StringBuilder in C#

`StringBuilder` is defined in the `System.Text` namespace. It is mutable and efficient for frequent string modifications.

```
using System.Text;

StringBuilder sb = new StringBuilder("Hello");
sb.Append(" World");
sb.Insert(5, ",");
sb.Replace("World", "C#");
sb.Remove(0, 1);
```

```
string result = sb.ToString();
```

3. String vs StringBuilder

Comparison based on key characteristics:

Feature	String	StringBuilder
Mutability	Immutable	Mutable
Performance	Slower for modifications	Faster for modifications
Thread Safety	Thread-safe	Not thread-safe
Use Case	Few modifications	Frequent modifications

Summary

Use `string` when your content is not expected to change frequently, and you value readability and simplicity. Use `StringBuilder` when you are performing multiple or complex string operations, especially in loops or large-scale text processing.