Most apps use strings. Best practices for working with strings.

Type string holds textual values.

“this is a string literal” – strings in C# are immutable meaning the value cannot be changed after it’s created. If a string is seemingly changed, a new string object is created.

It’s a ref type. Stores refs to it’s data. But we work with them as if they’re value types.

.NET provides an extensive library for string methods because most applications deal with strings.

E.G.

string.ToLower()

string.ToUpper()

string.Replace(“Vendor”, “Supplier”) – replaces any occurrence of 1st arg with 2nd

string.length;

string.IdexOf(“a”)

string.EndsWith(“world”)

string.StartsWith(“hello”)

NOTE: modifying methods don’t change the original string but will output a modified version of the string.

F10 – vis studio debug mode will execute a line from a breakpoint.

Avoid calling string methods on null strings.

Handling null strings:

String.IsNullOrWhiteSpace(theString) – static method returns true if null or whitespace.

Which is best null checking method?

If there’s many checks, enclose within an if and use IsNullOrWhiteSpace. If there’s only one or 2, use the nullable checker in C#6.

DO HANDLE NULL STRINGS!

DO WRITE UNIT TESTS TO COVER NULL STRINGS!

Verbatum string literals:

Tells C# not to process any backslash characters in the string. Defined by @ before the definition.

To apply backslash features into a verbatim string, add thm directly.

Environment.newLine = \n

DO USE VERBATIM STRING LITERALS TO HOLD PATH NAMES!

USE 2 QUOTES IN A VERBATIM TO INCLUDE QUOTES IN THE STRING “”a””!

Formatting strings:

String.Format(“{0}-{1}”, string1, string2) – creates a string along the lines of “string1 – string2”

The args can be other data types that call the ToString method of that object.

There is a better way in C#6 than String.Format.

String Interpolation (C#6 only. VisStudio 2015):

Lets us format strings easily.

Var p => $”{var1} – {var2}” – dollar sign notifies C# its an interpolated string to directly put the vars into the string literal.

Vars can be formatted within placeholders using a colon: {date:d}. formats the string to a date.

Building long strings:

StringBuilder allows strings to be modified without creating a new string.

StringBuilder is more efficient when concatenating many strings or modifying a string many times. Make the variable a StringBuilder type. Whenever you need the actual string call ToString on the StringBuilder.

There’s process and overhead in using StringBuilder so avoid using it if the string is only modified a few times.

FAQs

1. What does it mean to say C# strings are immutable?

It means once they are defined, they cannot be changed.

1. Is a string a value type or a reference type?

It’s a ref type but it’s used as a value type.

1. What’s the best way to determine if a string is null?

IsNullOrWhiteSpace or use null conditional operator.

1. What are the benefits to using StringBuilder?

Efficient in concatenating many strings together and if a string needs to be modified many times.