String.Empty VS ""

A brief overview on string interning and the compilation process in C#

BACKGROUND

In the past few months several fellow C# developers advised me that to create **empty string variables**, I should use **String.Empty** (or the lower-cased alias “string.Empty”) because each time **“”** is used, a new object is created, leading to performance penalties. **Is this accurate?** Here’s my investigation about the topic ...

SHORT ANSWER

No, it’s not!

Performance-wise there's no difference between the two. The reason for that is because of **string interning**. Both strings get interned. **How does that work?**

AN IN-DEPTH EXPLANATION

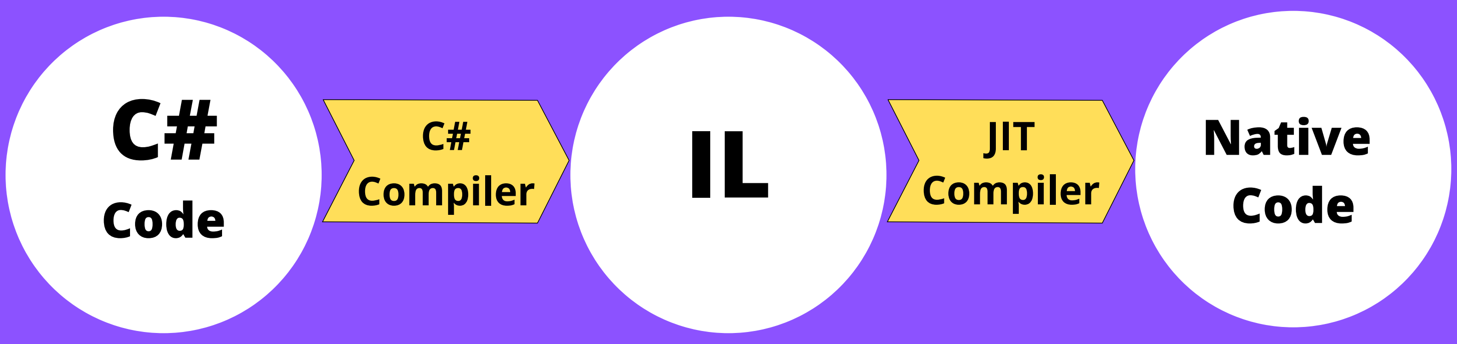
String Interning: CLR and the Intern Pool

Briefly, the **Common Language Runtime** stores strings by using a table called “**Intern Pool**” that contains a single reference to each **unique** literal string declared or created in your program. Because of that, an instance of a literal string with a particular value **only exists once** in the system. Consequently, all occurrences of **""** refer to the same string literal (for an empty string) as **string.Empty**, which means they are, in fact, equivalent.

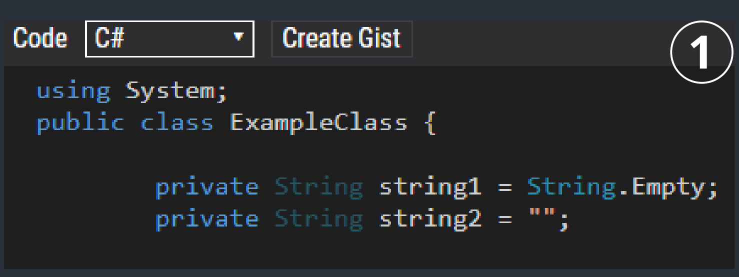
How can we test this?

THE COMPILATION PROCESS

Let's check a brief summary of how the compilation process works ...



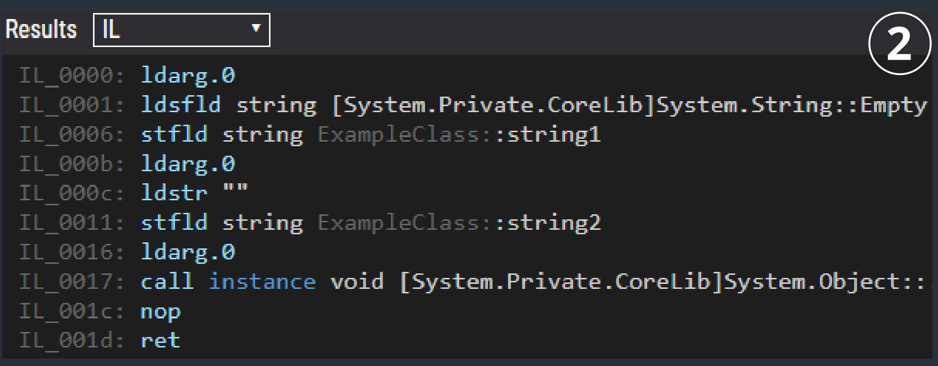
Here are some screen captures of this process, testing the creation of empty strings variables using **string.Empty** and **""**:



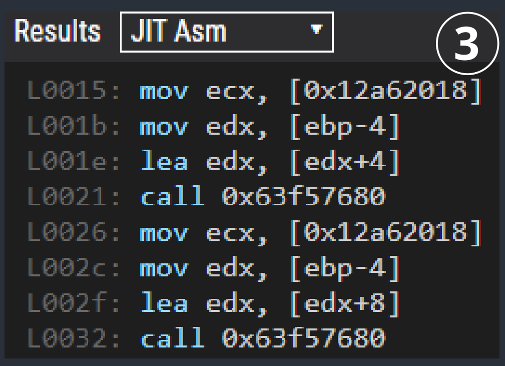
```
using System;
public class ExampleClass {

        private String string1 = String.Empty;
        private String string2 = "";
```

The source code written in **C#** **(1)** gets translated to **IL** (Intermediate Language) **(2)** and is later converted to **Native Code** (a CPU-specific code) by a **JIT** (just-in-time compiler) **(3).**



```
IL_0000: ldarg.0
IL_0001: ldsfld string [System.Private.CoreLib]System.String::Empty
IL_0006: stfld string ExampleClass::string1
IL_000b: ldarg.0
IL_000c: ldstr ""
IL_0011: stfld string ExampleClass::string2
IL_0016: ldarg.0
IL_0017: call instance void [System.Private.CoreLib]System.Object::
IL_001c: nop
IL_001d: ret
```



```
L0015: mov ecx, [0x12a62018]
L001b: mov edx, [ebp-4]
L001e: lea edx, [edx+4]
L0021: call 0x63f57680
L0026: mov ecx, [0x12a62018]
L002c: mov edx, [ebp-4]
L002f: lea edx, [edx+8]
L0032: call 0x63f57680
```

Without having to know exactly what every code line does, we can still easily see that in the final **JIT** output (3) the same memory address **[0x12a62018]** is used for both **string.Empty** and **""** that is moved using the "**mov**" statement into the register **ecx. This demonstrates that both are interned, meaning that they are practically the same.**

You can prove this by modifying one of the strings (adding some characters) in the C# code and then checking that, in this case, the memory addresses in the generated native code are going to differ.

MAKING A DECISION

Now that we know all of that, you may ask: Which one should I use?
Some would say that **""** is more concise, thus better for readability. Others may think that it can be confused with a string that a programmer forgot to complete, while string.Empty shows "intent" of it being really empty.

My answer: ? Choose either as long as you use it consistently throughout the code.
Disclaimer: The wrong fact that an object is created each time "" is used may date back to the initial versions of C#/.NET, where it may have worked that way (I couldn't find this specific behavior in the official documentation by Microsoft for those versions).

However, before wrapping this up, a few more thoughts ...

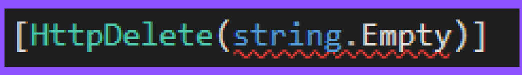
FINDING DIFFERENCES

One may think that it's possible to use them in any situation interchangeably, but, beware! There are some differences:

s***tring.Empty*** is a **readonly** field while **""** is a **const** meaning that the first one, as a readonly field, won't be suitable to be used in certain code blocks.

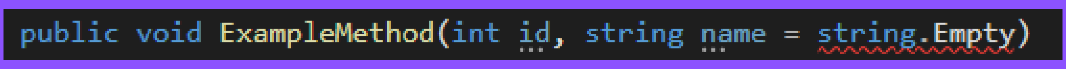
Some examples with code snippets (where string.Empty is **NOT** suitable) are:

- In **Attribute arguments**. *Example*:



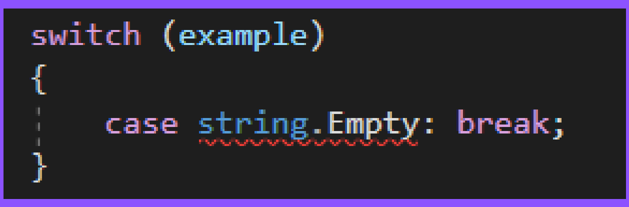
```
[HttpDelete(string.Empty)]
```

- As a ***default parameter.*** *Example:*



```
public void ExampleMethod(int id, string name = string.Empty)
```

- As a **case** expression in a **switch** statement. *Example:*



```
switch (example)
{
    case string.Empty: break;
}
```

REFERENCES AND LINKS

C#, IL, JIT prints generated at https://sharplab.io/q

Managed Execution Process (Microsoft Docs) at
https://docs.microsoft.com/en-us/dotnet/standard/managed-execution-process

String.Intern(String) Method (System) (Microsoft Docs) *at*
https://docs.microsoft.com/en-us/dotnet/api/system.string.intern?view=net-5.0

C# version used: 9.0

ABOUT

Hi, I'm Nahuel Ramos, a Software Analyst. I currently work as a developer at Softensity, back-end focused with .NET Core as the main technology. As a fan of the IT world, I'm constantly seeking to learn about new technologies and to grow as a professional.