How can I retrieve the namespace to a string C#

Asked 6 years, 2 months ago Active 2 years, 2 months ago Viewed 56k times

48

3

I am writing a program which needs the namespace of the program but I cant seem to figure out how to retrieve it. I would like the end result to be in a string.

I was able to find an MSDN page about this topic but it proved to be unhelpful to myself. http://msdn.microsoft.com/en-us/library/system.type.namespace.aspx

Any help would be appreciated. The program is written in C#.

EDIT: Sorry guys, this is not a console application.

c# string namespaces

asked Aug 28 '13 at 10:32
Elliot Ames
289 1 6 8

1 I don't understand clearly your question. Why don't you just write the namespace of your program where you want? Is it somehow changing? – IllidanS4 wants Monica back Aug 28 '13 at 10:41

8 @IllidanS4: This is a very valid question. Hard-coding the namespace in a string in the program is a recipe for disaster - sooner or later someone will change the namespace and forget or be unaware that it is also encoded in the program, and then it will fail. – RenniePet Aug 18 '14 at 2:46

1 Use case: embedded resources in an assembly have the path prefixed by the namespace. The answer below gives a strongly typed way of finding that prefix. – Tim Abell Jan 12 '17 at 13:36

9 Answers

83

This should work:

```
var myType = typeof(MyClass);
var n = myType.Namespace;
```

✓

Write out to the console:

```
Type myType = typeof(MyClass);
Console.WriteLine("Namespace: {0}.", myType.Namespace);
```

Setting a WinForm label:

```
Type myType = typeof(MyClass);
namespaceLabel.Text = myType.Namespace;
```

Or create a method in the relevant class and use anywhere:

```
public string GetThisNamespace()
{
   return GetType().Namespace;
}
```

edited Aug 26 '17 at 14:54

answered Aug 28 '13 at 10:35
Joe Ratzer
15.5k 2 32 46

It's unfortunately not a console application so the significance of the {0} in the line: Console.WriteLine(" Namespace: {0}.", myType.Namespace); is unknown to me. – Elliot Ames Aug 28 '13 at 10:35

Ok, what type of application is it? – Joe Ratzer Aug 28 '13 at 10:37

8 The WriteLine is not the essential part her. Use `string.Format()` and move on. – Henk Holterman Aug 28 '13 at 10:41

It's a windows forms application. – Elliot Ames Aug 28 '13 at 10:41

@ElliotAmes `myType.Namespace` in this case is a string – V4Vendetta Aug 28 '13 at 10:41

15

Put this to your assembly:

```
public static string GetCurrentNamespace()
{
    return
System.Reflection.Assembly.GetExecutingAssembly().EntryPoint.DeclaringType.Namespace;
}
```

Or if you want this method to be in a library used by your program, write it like this:

```
[System.Runtime.CompilerServices.MethodImpl(MethodImplOptions.NoInlining)]
public static string GetCurrentNamespace()
{
    return
System.Reflection.Assembly.GetCallingAssembly().EntryPoint.DeclaringType.Namespace;
}
```

edited Aug 28 '13 at 10:44

answered Aug 28 '13 at 10:37

IllidanS4 wants Monica back

7,358 2 33 62

8

To add to all the answers.

Since C# 6.0 there is the nameof keyword.

```
string name = nameof(MyNamespace);
```

This has several advantages:

1. The name is resolved at compile-time
2. The name will change when refactoring the namespace
3. It is syntax checked, so the name must exist
4. cleaner code

answered Jul 26 '16 at 8:12

Serve Laurijssen

5,781 4 25 56

10 This doesn't give the full namespace though. Would be great if it did. – The Pademelon Aug 10 '16 at 6:15

As it currently stands, this answer is misleading as it will *only* work with single segment namespaces. – julealgon Aug 12 at 17:04

8

if you have item `x` of class `A` in namespace `B` you can use:

```
string s = x.GetType().Namespace;
```

no `s` contains "B"

you can also use `x.GetType().Name` to get the type name or `x.GetType().FullName` to get both

answered Aug 28 '13 at 10:37

No Idea For Name

9,741 10 36 58

7

This can't go wrong:

```
MethodBase.GetCurrentMethod().DeclaringType.Namespace
```

answered Oct 3 '15 at 12:49

balage

381 7 8

1 How is this better than the accepted answer and the other variants posted here? – Nate Barbettini Oct 3 '15 at 16:38

3

You could simply use typeof and then pass in the class (I.e. Program):

```
Console.WriteLine(typeof(Program).Namespace);
```

Which would print:

```
ConsoleApplication1
```

answered Aug 28 '13 at 10:35

Darren

57.4k 18 110 122

▲

2

▼

```
Type myType = typeof(MyClass);
// Get the namespace of the myClass class.
Console.WriteLine("Namespace: {0}.", myType.Namespace);
```

Building on Joe's comment you can still use

```
Type myType = typeof(MyClass);
// Get the namespace of the myClass class.
var namespaceName = myType.Namespace.ToString();
```

with `namespaceName` being a variable to access the namespace name as a string value.

answered Aug 28 '13 at 10:38

BenM

3,228 2 23 53

▲

1

▼

as a roll upp all post answers: getting all columns' values from a table given as a string tableName:

```
     var tableName = "INVENTORY_PRICE";
                var assembly = Assembly.GetExecutingAssembly();

                var tip = typeof(Form3);

                var t = assembly.GetType(tip.Namespace + "." + tableName);
                if (t != null)
                {
                    var foos = db.GetTable(t);

                    foreach (var f in foos)
                    {
                        Console.WriteLine(f + ":");
                        foreach (var property in f.GetType().GetProperties())
                            if (property != null)
                            {
                                var pv = property.GetValue(f, null);
                                Console.WriteLine("   " + property.Name + ":" + pv);
                            }

                        Console.WriteLine("---------------------------------------------
---");
                    }
                }
```

it is very easy if we use ado, this sample uses LINQ context...

answered Jan 26 '17 at 3:18

Fatih ÜNAL

11 2

0

If you're executing it from a class in the namespace you need to capture then you can just use:

```
GetType().Namespace
```

This works nicely as it then allows you to refactor the namespace and will still work.

answered Sep 15 '16 at 9:51

Elliot Arazi

1