namespace

Article • 07/09/2022

The `namespace` keyword is used to declare a scope that contains a set of related objects. You can use a namespace to organize code elements and to create globally unique types.

C#

```
namespace SampleNamespace
{
    class SampleClass { }

    interface ISampleInterface { }

    struct SampleStruct { }

    enum SampleEnum { a, b }

    delegate void SampleDelegate(int i);

    namespace Nested
    {
        class SampleClass2 { }
    }
}
```

File scoped namespace declarations enable you to declare that all types in a file are in a single namespace. File scoped namespace declarations are available with C# 10. The following example is similar to the previous example, but uses a file scoped namespace declaration:

C#

```
using System;

namespace SampleFileScopedNamespace;

class SampleClass { }

interface ISampleInterface { }

struct SampleStruct { }

enum SampleEnum { a, b }

delegate void SampleDelegate(int i);
```

The preceding example doesn't include a nested namespace. File scoped namespaces can't include additional namespace declarations. You cannot declare a nested namespace or a second file-scoped namespace:

C#

```
namespace SampleNamespace;

class AnotherSampleClass
{
    public void AnotherSampleMethod()
    {
        System.Console.WriteLine(
            "SampleMethod inside SampleNamespace");
    }
}

namespace AnotherNamespace; // Not allowed!

namespace ANestedNamespace // Not allowed!
{
   // declarations...
}
```

Within a namespace, you can declare zero or more of the following types:

- class
- interface
- struct
- enum
- delegate
- nested namespaces can be declared except in file scoped namespace declarations

The compiler adds a default namespace. This unnamed namespace, sometimes referred to as the global namespace, is present in every file. It contains declarations not included in a declared namespace. Any identifier in the global namespace is available for use in a named namespace.

Namespaces implicitly have public access. For a discussion of the access modifiers you can assign to elements in a namespace, see Access Modifiers.

It's possible to define a namespace in two or more declarations. For example, the following example defines two classes as part of the `MyCompany` namespace:

C#

```
namespace MyCompany.Proj1
{
    class MyClass
    {
    }
}

namespace MyCompany.Proj1
{
    class MyClass1
    {
    }
}
```

The following example shows how to call a static method in a nested namespace.

C#

```
namespace SomeNameSpace
{
    public class MyClass
    {
        static void Main()
        {
            Nested.NestedNameSpaceClass.SayHello();
        }
    }

    // a nested namespace
    namespace Nested
    {
        public class NestedNameSpaceClass
        {
            public static void SayHello()
            {
                Console.WriteLine("Hello");
            }
        }
    }
}
// Output: Hello
```

C# language specification

For more information, see the Namespaces section of the C# language specification. For more information on file scoped namespace declarations, see the feature specification.

See also

- Namespace declaration preferences (IDE0160 and IDE0161)
- C# reference
- C# keywords
- using
- using static
- Namespace alias qualifier ::
- Namespaces