**Namespaces**

*Namespaces* are used to define scope in C# applications. By declaring a namespace, an application developer can give a C# application a hierarchical structure based on semantically related groups of types and other (nested) namespaces. Multiple source code files can contribute to the same namespace. To that extent, if you're packaging several classes in a given namespace, you can define each of these classes in its own source code file. A programmer employing your classes can get access to all the classes within a namespace through the *using* keyword.

## The *using* Keyword

Sometimes you'll want to use the fully qualified name for a given type in the form *namespace.type*. However, this can be rather tedious and sometimes isn't necessary. In the following example, we're using the *Console* object that exists in the *System* namespace.

class Using1

{

public static void Main()

{

**System.Console.WriteLine("test");**

}

}

However, what if we know that the *Console* object exists only in the *System* namespace? The *using* keyword allows us to specify a search order of namespaces so that when the compiler encounters a nonqualified type, it can look at the listed namespaces to search for the type. In the following example, the compiler locates the *Console* object in the *System* namespace without the developer having to specify it each time.

**using System;**

class Using2

{

public static void Main()

{

Console.WriteLine("test");

}

}

You cannot specify a class name with the *using* keyword. Therefore, the following would be invalid:

**using System.Console; // Invalid.**

class Using3

{

public static void Main()

{

WriteLine("test");

}

}

What you can do, however, is use a variation of the *using* keyword to create a *using* alias:

**using cc = System.Console;**

class Using4

{

public static void Main()

{

cc.WriteLine("test");

}

}

This is especially advantageous in cases where nested namespaces combined with long class names make it tedious to write code