**sealed (C# Reference)**

When applied to a class, the sealed modifier prevents other classes from inheriting from it. In the following example, class B inherits from class A, but no class can inherit from class B.

C#

class A {}

sealed class B : A {}

You can also use the sealed modifier on a method or property that overrides a virtual method or property in a base class. This enables you to allow classes to derive from your class and prevent them from overriding specific virtual methods or properties.

**Using type dynamic (C# Programming Guide)**

C# 4 introduces a new type, dynamic. The type is a static type, but an object of type dynamic bypasses static type checking. In most cases, it functions like it has type object. At compile time, an element that is typed as dynamic is assumed to support any operation. Therefore, you do not have to be concerned about whether the object gets its value from a COM API, from a dynamic language such as IronPython, from the HTML Document Object Model (DOM), from reflection, or from somewhere else in the program. However, if the code is not valid, errors are caught at run time.