<https://www.geeksforgeeks.org/custom-attributes-in-c-sharp/#:~:text=Attributes%20are%20metadata%20extensions%20that,of%20a%20piece%20of%20code>.

<https://www.geeksforgeeks.org/what-is-reflection-in-c-sharp/>

#define DEBUG

using System;

using System.Diagnostics;

public class Myclass {

[Conditional("DEBUG")]

public static void Message(string msg) {

Console.WriteLine(msg);

}

}

class Test {

static void function1() {

Myclass.Message("In Function 1.");

function2();

}

static void function2() {

Myclass.Message("In Function 2.");

}

public static void Main() {

Myclass.Message("In Main function.");

function1();

Console.ReadKey();

}

}

using System;

public class MyClass {

[Obsolete("Don't use OldMethod, use NewMethod instead", true)]

static void OldMethod() {

Console.WriteLine("It is the old method");

}

static void NewMethod() {

Console.WriteLine("It is the new method");

}

public static void Main() {

OldMethod();

}

}

[AttributeUsage(AttributeTargets.All)]

public class DeveloperInfoAttribute : Attribute

{

private string \_emailAddress;

private int \_revision;

public DeveloperInfo(string emailAddress, int revision)

{

this.\_emailAddress = emailAddress;

this.\_revision = revision;

}

public string EmailAddress

{

get { return this.\_emailAddress; }

}

public int Revision

{

get { return this.\_revision; }

}

}

Using a custom attribute is no different from using an attribute that the .NET Framework provides. You

simply apply the attribute to an element and ensure that you pass the required information to the

constructor.

The following code example shows how to apply the **DeveloperInfo** attribute to a type definition.

**Applying a Custom Attribute**

[DeveloperInfo("holly@fourthcoffee.com", 3)]

public class SalePerson

{

...

[DeveloperInfo("linda@fourthcoffee.com", 1)]

public IEnumerable<Sale> GetAllSales()

{

**MCT USE ONLY.**