System.Diagnostics.ConditionalAttribute Class

[ILAsm]

```
.class public sealed serializable ConditionalAttribute extends
System.Attribute
```

[C#]

```
public sealed class ConditionalAttribute: Attribute
```

Assembly Info:

- *Name:* mscorlib
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- *Version:* 2.0.x.x
- *Attributes:*
    - CLSCompliantAttribute(true)

Type Attributes:

- AttributeUsageAttribute(AttributeTargets.Method, AllowMultiple=true, Inherited=true)

Summary

Indicates to compilers that a method is callable if and only if a specified pre-processing identifier has been defined on the method.

Inherits From: System.Attribute

Library: BCL

Thread Safety: All public static members of this type are safe for multithreaded operations. No instance members are guaranteed to be thread safe.

Description

[*Note:* A `System.Diagnostics.ConditionalAttribute`, which has an associated condition `System.Diagnostics.ConditionalAttribute.ConditionString`, can be attached to the definition of a method, creating a *conditionalmethod*. Thereafter, when a compiler encounters a call to that method, it might choose to ignore the call unless a compilation variable is defined at the site of the call, with a value that matches in a case-sensitive manner the `System.Diagnostics.ConditionalAttribute.ConditionString` supplied to the `System.Diagnostics.ConditionalAttribute`.

Note that compilers might provide several techniques to define such compilation variables, such as:

- compiler command-line switches (for example, `/define:DEBUG`)
- environment variables in the operating system shell (for example, `SET DEBUG=1`)
- as pragmas in the source code (for example, `#define DEBUG`, to define the compilation variable, or `#undef DEBUG` to undefine it)

CLS-Compliant compilers are permitted to ignore uses of the `System.Diagnostics.ConditionalAttribute`.

]

Example

The following example demonstrates the use of `System.Diagnostics.ConditionalAttribute` with a particular compiler that supports the use of this attribute. The `System.Diagnostics.ConditionalAttribute.ConditionString` property of the current attribute is initialized as "DEBUG".

[C#]

```
using System;
using System.Diagnostics;

public class MyClass {

  [ConditionalAttribute("DEBUG")]
  public static void Display() {

    Console.WriteLine("Compiled with DEBUG");
  }
}

public class TestCondition {

  public static void Main() {

    Console.WriteLine("How was this compiled?");
    MyClass.Display();
    Console.WriteLine("<eop>");
  }
}
```

When this code is compiled with the compilation-variable DEBUG defined at the callsite, the output when run is

```
How was this compiled?


Compiled with DEBUG
```

```
<eop>
```

When this code is compiled without the compilation-variable DEBUG defined at the callsite, the output when run is

```
How was this compiled?
```

```
<eop>
```

ConditionalAttribute(System.String) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(string
conditionString)

[C#]
public ConditionalAttribute(string conditionString)
```

Summary

Constructs and initializes a new instance of the `System.Diagnostics.ConditionalAttribute` class.

Parameters

Parameter	Description
conditionString	A `System.String` that contains the pre-processing identifier that makes callable the target method of the current instance.

Description

This constructor initializes the `System.Diagnostics.ConditionalAttribute.ConditionString` property of the current instance using *conditionString*.

ConditionalAttribute.ConditionString Property

[ILAsm]

```
.property string ConditionString { public hidebysig specialname instance
string get_ConditionString() }
```

[C#]

```
public string ConditionString { get; }
```

Summary

Gets the `System.String` that contains the pre-processing identifier that makes callable the target method of the current instance.

Property Value

A `System.String` that contains the pre-processing identifier that makes callable the target method of the current instance.

Description

This property is read-only.