

# System.Reflection.MemberInfo Class

```
[ILAsm]
.class public abstract serializable MemberInfo extends System.Object

[C#]
public abstract class MemberInfo
```

## 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)

## Summary

Provides access to member metadata.

## Inherits From: System.Object

**Library:** Reflection

**Thread Safety:** This type is safe for multithreaded operations.

## Description

[*Note:* System.Reflection.MemberInfo is used to represent all members of a type: nested types, fields, events, properties, methods, and constructors. The Base Class Library includes the following derived types:

- System.Reflection.FieldInfo
- System.Reflection.EventInfo
- System.Reflection.PropertyInfo
- System.Type

]

# 1 MemberInfo() Constructor

```
2 [ILAsm]  
3 family rtspecialname specialname instance void .ctor()  
  
4 [C#]  
5 protected MemberInfo()
```

## 6 Summary

7 Constructs a new instance of the `System.Reflection.MemberInfo` class.

8

# MemberInfo.DeclaringType Property

```
[ILAsm]  
.property class System.Type DeclaringType { public hidebysig virtual  
abstract specialname class System.Type get_DeclaringType() }  
  
[C#]  
public abstract Type DeclaringType { get; }
```

## Summary

Gets the type that declares the member reflected by the current instance.

## Property Value

The `System.Type` object of the class that declares the member reflected by the current instance; or, `null` if the member reflected by the current instance is a global member.

## Description

[*Note:* A member of a class (or interface) is either declared on that type or inherited from a base class (or interface). The `System.Reflection.MemberInfo.DeclaringType` property value cannot be the same as the `System.Type` object used to obtain the current instance. These values will differ if either of the following conditions is true.

- If the `System.Type` object from which the current instance was obtained did not declare the member reflected by the current instance, the `System.Reflection.MemberInfo.DeclaringType` will represent the base type that is closest to that object in its hierarchy chain and declares the member reflected by the current instance.
- If the current instance reflects a global member, (that is, it was obtained from `System.Reflection.Module.GetMethods`, which returns global methods on a module), then the `System.Reflection.MemberInfo.DeclaringType` property value is `null`.

]

## Behaviors

This property is read-only.

This property is required to return the `System.Type` object for the type that declares the member reflected by the current instance. This property value is required to be equal to the `System.Reflection.MemberInfo.ReflectedType` property value of the current instance if and only if the reflected type also contains a declaration for the member reflected by the current instance.

## How and When to Override

1       Override this property to get the `System.Type` of the class that declared the member  
2       that is reflected by the current instance.

3

## 4   **Usage**

5       Use this property to determine the `System.Type` of the class that declared the member  
6       that is reflected by the current instance.

7

## 8   **Example**

9       The following example demonstrates the difference between the  
10       `System.Reflection.MemberInfo.DeclaringType` and  
11       `System.Reflection.MemberInfo.ReflectedType` of a member.

12       [C#]

```
14   using System;
15   using System.Reflection;
16
17   public class BaseClass {
18       public void ReflectedMethod() {}
19   }
20
21   public class DerivedClass: BaseClass {}
22
23   public class DeclaringTypeExample {
24       public static void Main() {
25           Type t = typeof(DerivedClass);
26           MemberInfo [] memInfo = t.GetMember("ReflectedMethod");
27           Console.WriteLine("Reflected type is {0}.", memInfo[0].ReflectedType);
28           Console.WriteLine("Declaring type is {0}.", memInfo[0].DeclaringType);
29       }
30   }
```

31       The output is

32       Reflected type is DerivedClass.

33

34

35       Declaring type is BaseClass.

36

37

# MemberInfo.Name Property

```
[ILAsm]
.property string Name { public hidebysig virtual abstract specialname
string get_Name() }

[C#]
public abstract string Name { get; }
```

## Summary

Gets the name of the member reflected by the current instance.

## Property Value

A `System.String` containing the name of the member reflected by the current instance.

## Behaviors

This property is read-only.

Only the simple name, not the fully qualified name, of the member reflected by the current instance is returned.

[*Note:* For example, if the current instance reflects the member `Print` in `System.MyClass`, the `System.Reflection.MemberInfo.Name` property would be `"Print"`.]

# MemberInfo.ReflectedType Property

```
[ILAsm]  
.property class System.Type ReflectedType { public hidebysig virtual  
abstract specialname class System.Type get_ReflectedType() }  
  
[C#]  
public abstract Type ReflectedType { get; }
```

## Summary

Gets the type of the class through which the current instance was obtained.

## Property Value

The `System.Type` object for the class through which the current instance was obtained.

## Behaviors

This property is read-only.

`ReflectedType` is required to get the type of the object that was used to obtain the current instance. This property value is required to be equal to the `System.Reflection.MemberInfo.DeclaringType` property value of the current instance if and only if the reflected type also contains a declaration for the member reflected by the current instance.