

System.Reflection.EventInfo Class

```
[ILAsm]
.class public abstract EventInfo extends System.Reflection.MemberInfo

[C#]
public abstract class EventInfo: 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 event metadata.

Inherits From: System.Reflection.MemberInfo

Library: Reflection

Thread Safety: This type is safe for multithreaded operations.

Description

Events are handled by delegates. An event listener supplies an event-handler delegate that is invoked whenever the event is raised by an event source. In order to connect to the event source, the event listener adds this delegate to the invocation list of the source. When the event is raised, the event-handler delegate invokes the methods in its invocation list. The `System.Reflection.EventInfo.GetAddMethod`, `System.Reflection.EventInfo.AddEventHandler`, `System.Reflection.EventInfo.GetRemoveMethod`, and `System.Reflection.EventInfo.RemoveEventHandler` methods, and the delegate type of the event-handler associated with an event, are required to be marked in the metadata.

[*Note:* For information on delegates, see the `System.Delegate` class overview.]

[*Note:* For information on events, see Partitions I and II of the CLI specification.]

1 **EventInfo() Constructor**

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

6 **Summary**

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

8

EventInfo.AddEventHandler(System.Object, System.Delegate) Method

```
[ILAsm]  
.method public hidebysig instance void AddEventHandler(object target,  
class System.Delegate handler)  
  
[C#]  
public void AddEventHandler(object target, Delegate handler)
```

Summary

Adds the specified event handler delegate to the specified event source.

Parameters

Parameter	Description
<i>target</i>	An object that represents an event source.
<i>handler</i>	A <code>System.Delegate</code> instance to be added to <i>target</i> that references methods to be invoked when the event reflected by the current instance is raised by <i>target</i> .

Description

Each time the event reflected by the current instance is raised by *target*, the methods in the invocation list of *handler* are invoked.

Exceptions

Exception	Condition
System.ArgumentException	<i>handler</i> is not the same <code>System.Type</code> as the event handler delegate declared for the event reflected by the current instance.
System.Reflection.TargetException	The event reflected by the current instance is non-static, and <i>obj</i> is <code>null</code> or is of a type that does not implement the event reflected by the current instance.

EventInfo.GetAddMethod(System.Boolean) Method

```
[ILAsm]  
.method public hidebysig virtual abstract class  
System.Reflection.MethodInfo GetAddMethod(bool nonPublic)  
  
[C#]  
public abstract MethodInfo GetAddMethod(bool nonPublic)
```

Summary

Returns the method used to add an event handler delegate to an event source for the event reflected by the current instance, specifying whether or not to return non-public methods.

Parameters

Parameter	Description
<i>nonPublic</i>	A System.Boolean value that specifies whether non-public methods can be returned by this method. Specify true to return non-public methods; otherwise, specify false.

Return Value

A System.Reflection.MethodInfo instance that reflects the method used to add an event handler delegate to an event source for the event reflected by the current instance, if found; otherwise, returns null.

Description

[Note: The returned method is used to add an event-handler delegate to the invocation list of an event source. Typically, the method has the following signature format:

```
add_<EventName>(<EventHandlerType> handler)  
]
```

Behaviors

As described above.

Exceptions

Exception	Condition
System.MethodAccessException	<i>nonPublic</i> is <code>true</code> , the method used to add an event handler delegate is non-public, and the caller does not have permission to reflect on non-public methods.

1

2 Permissions

Permission	Description
System.Security.Permissions.ReflectionPermission	Requires permission to reflect non-public members of a type in loaded assemblies. See <code>System.Security.Permissions.ReflectionPermissionFlag.TypeInformation</code> .

3

4

1 **EventInfo.GetAddMethod()** Method

```
2    [ILAsm]  
3    .method public hidebysig instance class System.Reflection.MethodInfo  
4    GetAddMethod()  
  
5    [C#]  
6    public MethodInfo GetAddMethod()
```

7 **Summary**

8 Returns the public method used to add an event handler delegate to an event source for
9 the event reflected by the current instance.

10 **Return Value**

11 A System.Reflection.MethodInfo instance that reflects the public method used to add
12 an event handler delegate to an event source for the event reflected by the current
13 instance, if found; otherwise, returns null.

14 **Description**

15 This method is equivalent to System.Reflection.EventInfo.GetAddMethod(false).

16
17 [*Note:* The returned method is used to add an event-handler delegate to the invocation
18 list of an event source. Typically, the method has the following signature format:

```
19       add_<EventName>(<EventHandlerType> handler)  
20  
21       ]  
22
```

MethodInfo.GetRaiseMethod(System.Boolean) Method

```
[ILAsm]  
.method public hidebysig virtual abstract class  
System.Reflection.MethodInfo GetRaiseMethod(bool nonPublic)  
  
[C#]  
public abstract MethodInfo GetRaiseMethod(bool nonPublic)
```

Summary

Returns the method that is called when the event reflected by the current instance is raised, specifying whether the method to be returned is public or non-public.

Parameters

Parameter	Description
<i>nonPublic</i>	A <code>System.Boolean</code> value that specifies whether non-public methods can be returned by this method. Specify <code>true</code> to return non-public methods; otherwise, specify <code>false</code> .

Return Value

A `System.Reflection.MethodInfo` instance that reflects the method that is called when the event reflected by the current instance is raised, if found; otherwise, returns `null`.

Behaviors

As described above.

Exceptions

Exception	Condition
System.MethodAccessException	<i>nonPublic</i> is <code>true</code> , the method used to raise the event is non-public, and the caller does not have permission to reflect on non-public methods.

Permissions

Permission	Description
System.Security.Permissions.ReflectionPermission	Requires permission to reflect non-public members of a type in loaded assemblies. See <code>System.Security.Permissions.ReflectionPermissionFlag.TypeInformation</code> .

1

2

EventInfo.GetRaiseMethod() Method

```
[ILAsm]  
.method public hidebysig instance class System.Reflection.MethodInfo  
GetRaiseMethod()  
  
[C#]  
public MethodInfo GetRaiseMethod()
```

Summary

Returns the public method that is called when the event reflected by the current instance is raised.

Return Value

A `System.Reflection.MethodInfo` instance that reflects the public method that is called when the event reflected by the current instance is raised, if found; otherwise, returns `null`.

EventInfo.GetRemoveMethod(System.Boolean) Method

```
[ILAsm]  
.method public hidebysig virtual abstract class  
System.Reflection.MethodInfo GetRemoveMethod(bool nonPublic)  
  
[C#]  
public abstract MethodInfo GetRemoveMethod(bool nonPublic)
```

Summary

Returns the method used to remove an event-handler delegate from the event reflected by the current instance, specifying whether or not to return non-public methods.

Parameters

Parameter	Description
<i>nonPublic</i>	A System.Boolean value that specifies whether non-public methods can be returned by this method. Specify true to return non-public methods; otherwise, specify false.

Return Value

A System.Reflection.MethodInfo instance that reflects the method used to remove an event handler delegate from the event reflected by the current instance, if found; otherwise, returns null.

Description

[Note: Typically, the method has the following signature format:

```
remove_<EventName>( <EventHandlerType> handler)  
]
```

Behaviors

As described above.

Exceptions

Exception	Condition
System.MethodAccessException	<i>nonPublic</i> is <code>true</code> , the method used to remove an event handler delegate is non-public, and the caller does not have permission to reflect on non-public methods.

1

2 Permissions

Permission	Description
System.Security.Permissions.ReflectionPermission	Requires permission to reflect non-public members of a type in loaded assemblies. See <code>System.Security.Permissions.ReflectionPermissionFlag.TypeInformation</code> .

3

4

1 **EventInfo.GetRemoveMethod() Method**

```
2    [ILAsm]  
3    .method public hidebysig instance class System.Reflection.MethodInfo  
4    GetRemoveMethod()  
  
5    [C#]  
6    public MethodInfo GetRemoveMethod()
```

7 **Summary**

8 Returns the public method used to remove an event-handler delegate from the event
9 reflected by the current instance.

10 **Return Value**

11 A System.Reflection.MethodInfo instance that reflects the public method used to
12 remove an event handler delegate from the event reflected by the current instance, if
13 found; otherwise, returns null.

14 **Description**

15 This method is equivalent to System.Reflection.EventInfo.GetRemoveMethod(false).

16
17 [*Note:* Typically, the method has the following signature format:

```
18       remove_<EventName>(<EventHandlerType> handler)  
19  
20  
21       ]
```

EventInfo.RemoveEventHandler(System.Object, System.Delegate) Method

```
[ILAsm]  
.method public hidebysig instance void RemoveEventHandler(object target,  
class System.Delegate handler)  
  
[C#]  
public void RemoveEventHandler(object target, Delegate handler)
```

Summary

Removes the specified event handler delegate from the specified event source.

Parameters

Parameter	Description
<i>target</i>	An object that represents an event source.
<i>handler</i>	A <code>System.Delegate</code> instance to be disassociated from the events reflected by the current instance that are raised by <i>target</i> .

Description

After this method is invoked, subsequent events reflected by the current instance that are raised by *target* will no longer cause *handler* to invoke its methods.

Exceptions

Exception	Condition
System.ArgumentException	<i>handler</i> is not the same type <code>System.Type</code> as the event handler delegate declared for the event reflected by the current instance.

1 EventInfo.Attributes Property

```
2 [ILAsm]  
3 .property valuetype System.Reflection.EventAttributes Attributes { public  
4 hidebysig virtual abstract specialname valuetype  
5 System.Reflection.EventAttributes get_Attributes() }  
  
6 [C#]  
7 public abstract EventAttributes Attributes { get; }
```

8 Summary

9 Gets the attributes of the event reflected by the current instance.

10 Property Value

11 A System.Reflection.EventAttributes value that specifies the attributes in the
12 metadata of the event reflected by the current instance.

EventInfo.EventHandlerType Property

```
[ILAsm]
.property class System.Type EventHandlerType { public hidebysig
specialname instance class System.Type get_EventHandlerType() }

[C#]
public Type EventHandlerType { get; }
```

Summary

Gets the System.Type of the event-handler System.Delegate associated with the event reflected by the current instance.

Property Value

A System.Type that represents the type of the event-handler System.Delegate associated with the event reflected by the current instance. Returns null if the method used to add a delegate to the event is not public and is in a loaded assembly, and the caller does not have the required permission.

Description

This property is read-only.

Permissions

Permission	Description
System.Security.Permissions.ReflectionPermission	Requires permission to reflect non-public members of a type in loaded assemblies. See System.Security.Permissions.ReflectionPermissionFlag.TypeInformation.