

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 00 04 00 00 00 00 00 00]
- *Version:* 1.0.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**
8 class.

9

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

```
3 [ILASM]  
4 .method public hidebysig instance void  
5 AddEventHandler(object target, class System.Delegate  
6 handler)  
  
7 [C#]  
8 public void AddEventHandler(object target, Delegate  
9 handler)
```

10 **Summary**

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

13 **Parameters**

Parameter	Description
<i>target</i>	An object that represents an event source.
<i>handler</i>	A System.Delegate 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> .

16 **Description**

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

20 **Exceptions**

Exception	Condition
System.ArgumentException	<i>handler</i> is not the same type System.Type 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 null or is of a type that does not implement the event reflected by the current instance.

23
24
25

1 **MethodInfo.GetAddMethod(System.Boolean** 2 **) Method**

```
3 [ILASM]  
4 .method public hidebysig virtual abstract class  
5 System.Reflection.MethodInfo GetAddMethod(bool nonPublic)  
  
6 [C#]  
7 public abstract MethodInfo GetAddMethod(bool nonPublic)
```

8 **Summary**

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

12 **Parameters**

13
14

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 .

15
16
17

16 **Return Value**

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

22 **Description**

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

```
26  
27 add_<EventName>( <EventHandlerType> handler)]
```

28 **Behaviors**

29 As described above.

30 **Exceptions**

31
32

Exception	Condition
System.MethodAccessException	<i>nonPublic</i> is true , 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
3
4

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 .

5
6
7

1 **MethodInfo.GetAddMethod()** Method

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

7 **Summary**

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

10 **Return Value**

11

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

16 **Description**

17 This method is equivalent to
18 **System.Reflection.EventInfo.GetAddMethod(false)**.

19

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

23

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

25

1 EventInfo.GetRaiseMethod(System.Boolean) Method

```
3 [ILASM]  
4 .method public hidebysig virtual abstract class  
5 System.Reflection.MethodInfo GetRaiseMethod(bool nonPublic)  
  
6 [C#]  
7 public abstract MethodInfo GetRaiseMethod(bool nonPublic)
```

8 Summary

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

12 Parameters

13
14

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 .

15
16
17

16 Return Value

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

21 Behaviors

22 As described above.

23 Exceptions

24
25

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

26
27
28
29

27 Permissions

Permission	Description
------------	-------------

1
2
3

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

1 **EventInfo.GetRaiseMethod()** Method

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

7 **Summary**

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

10 **Return Value**

11

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

15

1 EventInfo.GetRemoveMethod(System.Boolean) Method

```
3 [ILASM]
4 .method public hidebysig virtual abstract class
5 System.Reflection.MethodInfo GetRemoveMethod(bool
6 nonPublic)
7
8 [C#]
9 public abstract MethodInfo GetRemoveMethod(bool nonPublic)
```

9 Summary

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

13 Parameters

14
15

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 .

16
17
18

17 Return Value

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

22 Description

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

24
25

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

26 Behaviors

27 As described above.

28 Exceptions

29
30

Exception	Condition
System.MethodAccessException	<i>nonPublic</i> is true , the method used to remove an event handler delegate is non-public, and the

1
2
3
4

Permissions

	caller does not have permission to reflect on non-public methods.
--	---

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 .

5
6
7

1 **MethodInfo.GetRemoveMethod()** Method

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

7 **Summary**

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

10 **Return Value**

11

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

15 **Description**

16 This method is equivalent to
17 **System.Reflection.EventInfo.GetRemoveMethod(false)**.

18

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

20

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

22

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

```
3 [ILASM]
4 .method public hidebysig instance void
5 RemoveEventHandler(object target, class System.Delegate
6 handler)
7
8 [C#]
9 public void RemoveEventHandler(object target, Delegate
10 handler)
```

10 Summary

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

13 Parameters

14
15

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

16
17

17 Description

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

21 Exceptions

22
23

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

24
25
26

1 EventInfo.Attributes Property

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

9 Summary

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

11 Property Value

12

13 A **System.Reflection.EventAttributes** value that specifies the
14 attributes in the metadata of the event reflected by the current
15 instance.

16

1 EventInfo.EventHandlerType Property

```
2 [ILASM]
3 .property class System.Type EventHandlerType { public
4 hidebysig specialname instance class System.Type
5 get_EventHandlerType() }
6
7 [C#]
8 public Type EventHandlerType { get; }
```

8 Summary

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

11 Property Value

12
13 A **System.Type** that represents the type of the event-handler
14 **System.Delegate** associated with the event reflected by the current
15 instance. Returns **null** if the method used to add a delegate to the
16 event is not public and is in a loaded assembly, and the caller does not
17 have the required permission.

18 Description

19 This property is read-only.

20 Permissions

21
22

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 .

23
24