

1 System.Security.Permissions.FileIOPermissionAttribute Class

```
3 [ILAsm]  
4 .class public sealed serializable FileIOPermissionAttribute extends  
5 System.Security.Permissions.CodeAccessSecurityAttribute  
  
6 [C#]  
7 public sealed class FileIOPermissionAttribute: CodeAccessSecurityAttribute
```

8 Assembly Info:

- 9 • *Name:* mscorlib
- 10 • *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- 11 • *Version:* 2.0.x.x
- 12 • *Attributes:*
 - 13 ○ CLSCompliantAttribute(true)

14 Type Attributes:

- 15 • AttributeUsageAttribute(AttributeTargets.Assembly | AttributeTargets.Class |
16 AttributeTargets.Struct | AttributeTargets.Constructor | AttributeTargets.Method,
17 AllowMultiple=true, Inherited=false)

18 Summary

19 Used to declaratively specify security actions to control access to files and directories.

20 Inherits From: System.Security.Permissions.CodeAccessSecurityAttribute

21
22 **Library:** BCL

23
24 **Thread Safety:** All public static members of this type are safe for multithreaded operations.
25 No instance members are guaranteed to be thread safe.

27 Description

28 [Note: The level of access to a file or directory is specified using the members of the
29 current instance. For example, to specify read permissions for a file, set the
30 System.Security.Permissions.FileIOPermissionAttribute.Read property equal to
31 the name of the file.

32
33 The security information declared by a security attribute is stored in the metadata of the
34 attribute target, and is accessed by the system at run-time. Security attributes are used
35 for declarative security only. For imperative security, use the corresponding permission
36 class, System.Security.Permissions.FileIOPermission.

37
38 The allowable System.Security.Permissions.FileIOPermissionAttribute targets are

1 determined by the `System.Security.Permissions.SecurityAction` passed to the
2 constructor.

3
4]

5
6 Case-sensitivity of file and directory names is platform dependent. The set of characters
7 that are valid for use in file and directory names is determined by the current file
8 system.

9 **Example**

10 The following example shows a declarative request for full access to the specified file.
11 The `System.Security.Permissions.SecurityAction.RequestMinimum` security action
12 indicates that this is the minimum permission required for the target assembly to be
13 able to execute.

14
15 `[assembly:FileIOPermissionAttribute(SecurityAction.RequestMinimum,`
16 `All="\\example\\sample.txt")]`

17
18 The following example shows how to demand that the calling code has unrestricted
19 access to files and directories. Demands are typically made to protect methods or
20 classes from malicious code.

21
22 `[FileIOPermissionAttribute(SecurityAction.Demand, Unrestricted=true)]`

23

1
2 **FileIOPermissionAttribute(System.Security.P**
3 **ermissions.SecurityAction) Constructor**

```
4 [ILAsm]  
5 public rtspecialname specialname instance void .ctor(valuetype  
6 System.Security.Permissions.SecurityAction action)  
  
7 [C#]  
8 public FileIOPermissionAttribute(SecurityAction action)
```

9 **Summary**

10 Constructs and initializes a new instance of the
11 System.Security.Permissions.FileIOPermissionAttribute class with the specified
12 System.Security.Permissions.SecurityAction value.

13 **Parameters**

Parameter	Description
<i>action</i>	A System.Security.Permissions.SecurityAction value.

14
15 **Exceptions**

Exception	Condition
System.ArgumentException	<i>action</i> is not a valid System.Security.Permissions.SecurityAction value.

16
17

1
2 **FileIOPermissionAttribute.CreatePermission(**
3 **) Method**

```
4 [ILAsm]  
5 .method public hidebysig virtual class System.Security.IPermission  
6 CreatePermission()  
7 [C#]  
8 public override IPermission CreatePermission()
```

9 **Summary**

10 Returns a new System.Security.Permissions.FileIOPermission that contains the
11 security information of the current instance.

12 **Return Value**

13 A new System.Security.Permissions.FileIOPermission object with the security
14 information of the current instance.

15 **Description**

16 [*Note:* Applications typically do not call this method; it is intended for use by the
17 system.

18
19 The security information declared by a security attribute is stored in the metadata of the
20 attribute target, and is accessed by the system at run-time. The system uses the object
21 returned by this method to convert the security information of the current instance into
22 the form stored in metadata.

23
24 This method overrides
25 System.Security.Permissions.SecurityAttribute.CreatePermission.

26
27]

28

1 FileIOPermissionAttribute.All Property

```
2 [ILAsm]  
3 .property string All { public hidebysig specialname instance void  
4 set_All(string value) }  
  
5 [C#]  
6 public string All { set; }
```

7 Summary

8 Sets the name of a file or directory for which full access is secured.

9 Property Value

10 A `System.String` containing the absolute path of the file or directory for which full
11 access is secured.

12 Description

13 This property is write-only.

14
15 [*Note:* This property sets full access for a single file or directory; use additional
16 `System.Security.Permissions.FileIOPermissionAttribute` attributes to specify
17 additional files and directories.]
18
19

20

1 FileIOPermissionAttribute.Append Property

```
2 [ILAsm]  
3 .property string Append { public hidebysig specialname instance string  
4 get_Append() public hidebysig specialname instance void set_Append(string  
5 value) }  
  
6 [C#]  
7 public string Append { get; set; }
```

8 Summary

9 Gets or sets the name of a file or directory for which append access is secured.

10 Property Value

11 A `System.String` containing the absolute path of the file or directory for which append
12 access is secured.

13 Description

14 [*Note:* This property sets append access for a single file or directory; use additional
15 `System.Security.Permissions.FileIOPermissionAttribute` attributes to specify
16 additional files and directories.]
17
18

19

1 FileIOPermissionAttribute.PathDiscovery 2 Property

```
3 [ILAsm]  
4 .property string PathDiscovery { public hidebysig specialname instance  
5 string get_PathDiscovery() public hidebysig specialname instance void  
6 set_PathDiscovery(string value) }  
  
7 [C#]  
8 public string PathDiscovery { get; set; }
```

9 Summary

10 Gets or sets the name of a file or directory for which path discovery access is secured.

11 Property Value

12 A `System.String` containing the absolute path of the file or directory for which access to
13 the contents of the path is secured.

14 Description

15 [*Note:* This property sets path discovery access for a single file or directory; use
16 additional `System.Security.Permissions.FileIOPermissionAttribute` attributes to
17 specify additional files and directories.

18
19 Path discovery controls access to the information in the path itself. This protects
20 sensitive information in the path, such as user names, as well as information about the
21 directory structure revealed in the path. This value does not secure access to files or
22 folders represented by the path.

23
24]

25

1 FileIOPermissionAttribute.Read Property

```
2 [ILAsm]  
3 .property string Read { public hidebysig specialname instance string  
4 get_Read() public hidebysig specialname instance void set_Read(string  
5 value) }  
  
6 [C#]  
7 public string Read { get; set; }
```

8 Summary

9 Gets or sets the name of a file or directory for which read access is secured.

10 Property Value

11 A `System.String` containing the absolute path of the file or directory for which read
12 access is secured.

13 Description

14 [*Note:* This property sets read access for a single file or directory; use additional
15 `System.Security.Permissions.FileIOPermissionAttribute` attributes to specify
16 additional files and directories.]
17
18

19

1 FileIOPermissionAttribute.Write Property

```
2 [ILAsm]  
3 .property string Write { public hidebysig specialname instance string  
4 get_Write() public hidebysig specialname instance void set_Write(string  
5 value) }  
  
6 [C#]  
7 public string Write { get; set; }
```

8 Summary

9 Gets or sets the name of a file or directory for which write access is secured.

10 Property Value

11 A `System.String` containing the absolute path of the file or directory for which write
12 access is secured.

13 Description

14 [*Note:* This property sets write access for a single file or directory; use additional
15 `System.Security.Permissions.FileIOPermissionAttribute` attributes to specify
16 additional files and directories.]
17
18

19