

1 System.Security.Permissions.EnvironmentPer 2 missionAttribute Class

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

9 Assembly Info:

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

15 Type Attributes:

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

19 Summary

20 Used to declaratively specify security actions to control access to environment variables.

21 Inherits From: System.Security.Permissions.CodeAccessSecurityAttribute

22
23 **Library:** BCL

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

27 28 Description

29 Environment variable names are case-insensitive. Multiple environment variable names
30 are specified by separating the names using the `System.IO.Path.PathSeparator`
31 string.

32
33 [*Note:* The level of access to one or more environment variables is specified using the
34 members of the current instance. For example, to specify read permissions for an
35 environment variable, set the
36 `System.Security.Permissions.EnvironmentPermissionAttribute.Read` property
37 equal to the name of the environment variable.
38

1 The security information declared by a security attribute is stored in the metadata of the
2 attribute target, and is accessed by the system at run-time. Security attributes are used
3 for declarative security only. For imperative security, use the corresponding permission
4 class, `System.Security.Permissions.EnvironmentPermission`.
5
6 The allowable `System.Security.Permissions.EnvironmentPermissionAttribute`
7 targets are determined by the `System.Security.Permissions.SecurityAction` passed
8 to the constructor.
9
10]

11 Example

12 The following example shows a declarative request for the ability to read the specified
13 environment variables. The
14 `System.Security.Permissions.SecurityAction.RequestMinimum` security action
15 indicates that this is the minimum permission required for the target assembly to be
16 able to execute.

```
17  
18 [assembly:EnvironmentPermissionAttribute(SecurityAction.RequestMinimum,  
19 Read="COMPUTERNAME;USERNAME;USERDOMAIN" ) ]  
20
```

21 The following example shows how to demand that the calling code has unrestricted
22 access to all environment variables. Demands are typically made in managed libraries to
23 protect methods or classes from malicious code.

```
24 [EnvironmentPermissionAttribute(SecurityAction.Demand, Unrestricted=true)]  
25
```

26

1
2 **EnvironmentPermissionAttribute(System.Security.Permissions.SecurityAction) Constructor**
3

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

9 **Summary**

10 Constructs and initializes a new instance of the
11 System.Security.Permissions.EnvironmentPermissionAttribute class with the
12 specified 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 EnvironmentPermissionAttribute.CreatePermi 3 ssion() 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.EnvironmentPermission that contains
11 the security information of the current instance.

12 Return Value

13 A new System.Security.Permissions.EnvironmentPermission object with the
14 security 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 described by a security attribute is stored in the metadata of
20 the attribute target, and is accessed by the system at run-time. The system uses the
21 object returned by this method to convert the security information of the current
22 instance into the form stored in metadata.

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

26
27]

28

1 EnvironmentPermissionAttribute.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 environment variables for which full access is secured.

9 Property Value

10 A `System.String` containing one or more environment variables for which full access is
11 secured.

12 Description

13 This property is write-only.

14
15 Multiple environment variable names are specified by separating the names using the
16 `System.IO.Path.PathSeparator` string. Environment variable names are case-
17 insensitive.

18
19 [*Note:* The security action passed to the constructor of the current instance determines
20 how the specified environment variables are secured. For example, if the action is
21 `System.Security.Permissions.SecurityAction.RequestMinimum`, then the target of
22 the current instance requires full access to the specified variables in order to execute. If
23 the action is `System.Security.Permissions.SecurityAction.RequestRefuse`, then
24 the system does not allow the target any access to the specified variables.]

25
26

27

1 EnvironmentPermissionAttribute.Read 2 Property

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

9 Summary

10 Gets or sets the environment variables for which read access is secured.

11 Property Value

12 A System.String containing one or more environment variables for which read access is
13 secured.

14 Description

15 Multiple environment variable names are specified by separating the names using the
16 System.IO.Path.PathSeparator string. Environment variable names are case-
17 insensitive.

18
19 [*Note:* The security action passed to the constructor of the current instance determines
20 how the specified environment variables are secured. For example, if the action is
21 System.Security.Permissions.SecurityAction.RequestMinimum, then the target of
22 the current instance requires read access to the specified variables in order to execute.
23 If the action is System.Security.Permissions.SecurityAction.RequestRefuse, then
24 the system does not allow the target to read the specified variables.]
25
26

27

1 EnvironmentPermissionAttribute.Write 2 Property

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

9 Summary

10 Gets or sets the environment variables for which write access is secured.

11 Property Value

12 A `System.String` containing one or more environment variables for which write access
13 is secured.

14 Description

15 Multiple environment variable names are specified by separating the names using the
16 `System.IO.Path.PathSeparator` string. Environment variable names are case-
17 insensitive.

18
19 [*Note:* The security action passed to the constructor of the current instance determines
20 how the specified environment variables are secured. For example, if the action is
21 `System.Security.Permissions.SecurityAction.RequestMinimum`, then the target of
22 the current instance requires write access to the specified variables in order to execute.
23 If the action is `System.Security.Permissions.SecurityAction.RequestRefuse`, then
24 the system does not allow the target to write the specified variables.]
25
26

27