System.Security.Permissions.EnvironmentPermissionAttribute Class

```
[ILAsm]
.class public sealed serializable EnvironmentPermissionAttribute
extends System.Security.Permissions.CodeAccessSecurityAttribute

[C#]
public sealed class EnvironmentPermissionAttribute:
CodeAccessSecurityAttribute
```

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)

Type Attributes:

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

Summary

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

Inherits From: System.Security.Permissions.CodeAccessSecurityAttribute

Library: BCL

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

Description

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

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

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

The allowable `System.Security.Permissions.EnvironmentPermissionAttribute` targets are determined by the `System.Security.Permissions.SecurityAction` passed to the constructor.

]

Example

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

```
[assembly:EnvironmentPermissionAttribute(SecurityAction.RequestMinim
um, Read="COMPUTERNAME;USERNAME;USERDOMAIN")]
```

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

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

EnvironmentPermissionAttribute(System.Security.Permissions.SecurityAction) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(valuetype
System.Security.Permissions.SecurityAction action)

[C#]
public EnvironmentPermissionAttribute(SecurityAction action)
```

Summary

Constructs and initializes a new instance of the `System.Security.Permissions.EnvironmentPermissionAttribute` class with the specified `System.Security.Permissions.SecurityAction` value.

Parameters

Parameter	Description
action	A `System.Security.Permissions.SecurityAction` value.

Exceptions

Exception	Condition
System.ArgumentException	*action* is not a valid `System.Security.Permissions.SecurityAction` value.

EnvironmentPermissionAttribute.CreatePermission() Method

```
[ILAsm]
.method public hidebysig virtual class System.Security.IPermission
CreatePermission()

[C#]
public override IPermission CreatePermission()
```

Summary

Returns a new `System.Security.Permissions.EnvironmentPermission` that contains the security information of the current instance.

Return Value

A new `System.Security.Permissions.EnvironmentPermission` object with the security information of the current instance.

Description

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

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

This method overrides `System.Security.Permissions.SecurityAttribute.CreatePermission`.

]

EnvironmentPermissionAttribute.All Property

```
[ILAsm]
.property string All { public hidebysig specialname instance void
set_All(string value) }

[C#]
public string All { set; }
```

Summary

Sets the environment variables for which full access is secured.

Property Value

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

Description

This property is write-only.

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

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

EnvironmentPermissionAttribute.Read Property

```
[ILAsm]
.property string Read { public hidebysig specialname instance string
get_Read() public hidebysig specialname instance void
set_Read(string value) }

[C#]
public string Read { get; set; }
```

Summary

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

Property Value

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

Description

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

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

EnvironmentPermissionAttribute.Write Property

```
[ILAsm]
.property string Write { public hidebysig specialname instance
string get_Write() public hidebysig specialname instance void
set_Write(string value) }

[C#]
public string Write { get; set; }
```

Summary

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

Property Value

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

Description

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

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