System.Security.Permissions.SecurityPermission Class

```
[ILAsm]
.class public sealed serializable SecurityPermission extends
System.Security.CodeAccessPermission

[C#]
public sealed class SecurityPermission: CodeAccessPermission
```

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)

Implements:

- **System.Security.IPermission**

Summary

Describes a set of security permissions applied to code.

Inherits From: System.Security.CodeAccessPermission

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

The `System.Security.Permissions.SecurityPermissionFlag` enumeration defines the permissions secured by this class.

The XML encoding of a `System.Security.Permissions.SecurityPermission` instance is defined below in EBNF format. The following conventions are used:

- All non-literals in the grammar below are shown in normal type.

- All literals are in bold font.

The following meta-language symbols are used:

- '*' represents a meta-language symbol suffixing an expression that can appear zero or more times.
- '?' represents a meta-language symbol suffixing an expression that can appear zero or one time.
- '+' represents a meta-language symbol suffixing an expression that can appear one or more times.
- '(',')' is be used to group literals, non-literals or a mixture of literals and non-literals.
- '|' denotes an exclusive disjunction between two expressions.
- '::= ' denotes a production rule where a left hand non-literal is replaced by a right hand expression containing literals, non-literals or both.

BuildVersion refers to the build version of the shipping CLI. This is a dotted build number such as '2412.0'.

ECMAPubKeyToken::= `b77a5c561934e089`

SecurityPermissionFlag = `Assertion` | `ControlThread` | `Execution` | `SkipVerification` | `UnmanagedCode`

Each SecurityPermissionFlag literal can appear in the XML no more than once. For example, `Flags=Assertion,Assertion` is illegal.

SecurityPermission::=

```
<IPermission

class="

System.Security.Permissions.SecurityPermission,

mscorlib,

Version=1.0.BuildVersion,

Culture=neutral,

PublicKeyToken=ECMAPubKeyToken"
```

```
version="1"

(

Unrestricted="true"

)

|

(

Flags="SecurityPermissionFlag (, SecurityPermissionFlag)* ")

| ()

/>
```

SecurityPermission(System.Security.Permissions.PermissionState) Constructor

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

[C#]
public SecurityPermission(PermissionState state)
```

Summary

Constructs a new instance of the `System.Security.Permissions.SecurityPermission` class with the specified `System.Security.Permissions.PermissionState` value.

Parameters

Parameter	Description
state	A `System.Security.Permissions.PermissionState` value. This value is either `System.Security.Permissions.PermissionState.None` or `System.Security.Permissions.PermissionState. Unrestricted`, respectively yielding fully restricted or fully unrestricted access to all security variables.

Exceptions

Exception	Condition
System.ArgumentException	*state* is not a valid `System.Security.Permissions.PermissionState` value.

SecurityPermission(System.Security.Permissions.SecurityPermissionFlag) Constructor

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

[C#]
public SecurityPermission(SecurityPermissionFlag flag)
```

Summary

Constructs a new instance of the `System.Security.Permissions.SecurityPermission` class with the specified `System.Security.Permissions.SecurityPermissionFlag` value.

Parameters

Parameter	Description
flag	One or more `System.Security.Permissions.SecurityPermissionFlag` values. Specify multiple values for *flag* using the bitwise OR operator.

Exceptions

Exception	Condition
System.ArgumentException	*flag* is not a valid `System.Security.Permissions.SecurityPermissionFlag` value.

SecurityPermission.Copy() Method

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

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

Summary

Returns a `System.Security.Permissions.SecurityPermission` object containing the same values as the current instance.

Return Value

A new `System.Security.Permissions.SecurityPermission` instance containing the same values as the current instance.

Description

[*Note:* The object returned by this method represents the same access to resources as the current instance.

This method overrides `System.Security.CodeAccessPermission.Copy` and is implemented to support the `System.Security.IPermission` interface.

]

SecurityPermission.FromXml(System.Security.SecurityElement) Method

```
[ILAsm]
.method public hidebysig virtual void FromXml(class
System.Security.SecurityElement esd)

[C#]
public override void FromXml(SecurityElement esd)
```

Summary

Reconstructs the state of a `System.Security.Permissions.SecurityPermission` object using the specified XML encoding.

Parameters

Parameter	Description
esd	A `System.Security.SecurityElement` instance containing the XML encoding to use to reconstruct the state of a `System.Security.Permissions.SecurityPermission` object.

Description

The state of the current instance is changed to the state encoded in *esd*.

[*Note:* For the XML encoding for this class, see the `System.Security.Permissions.SecurityPermission` class page.

This method overrides `System.Security.CodeAccessPermission.FromXml`.

]

Exceptions

Exception	Condition
System.ArgumentNullException	*esd* is `null`.
System.ArgumentException	*esd* does not contain the encoding for a `System.Security.Permissions.SecurityPermission` instance.

	The version number of *esd* is not valid.

SecurityPermission.Intersect(System.Security.IPermission) Method

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

[C#]
public override IPermission Intersect(IPermission target)
```

Summary

Returns a `System.Security.Permissions.SecurityPermission` object that is the intersection of the current instance and the specified object.

Parameters

Parameter	Description
target	A `System.Security.Permissions.SecurityPermission` object that is of the same type as the current instance to be intersected with the current instance.

Return Value

A new `System.Security.Permissions.SecurityPermission` instance that represents the intersection of the current instance and *target*. If the intersection is empty, or *target* is `null`, returns `null`.

Description

[*Note:* The intersection of two permissions is a permission that secures the resources and operations secured by both permissions. Specifically, it represents the minimum permission such that any demand that passes both permissions will also pass their intersection.

This method overrides `System.Security.CodeAccessPermission.Intersect` and is implemented to support the `System.Security.IPermission` interface.

]

Exceptions

Exception	Condition
System.ArgumentException	*target* is not `null` and is not of type

	System.Security.Permissions.SecurityPermission.

SecurityPermission.IsSubsetOf(System.Security.IPermission) Method

```
[ILAsm]
.method public hidebysig virtual bool IsSubsetOf(class
System.Security.IPermission target)

[C#]
public override bool IsSubsetOf(IPermission target)
```

Summary

Determines whether the current instance is a subset of the specified object.

Parameters

Parameter	Description
target	A `System.Security.Permissions.SecurityPermission` object of the same type as the current instance that is to be tested for the subset relationship with the current instance.

Return Value

`true` if the current instance is a subset of *target*; otherwise, `false`. If the current instance is unrestricted, and *target* is not, returns `false`. If *target* is unrestricted, returns `true`. If target is `null` and the current instance was initialized with `System.Security.Permissions.SecurityPermissionFlag.NoFlags`, returns `true`. If target is `null` and the current instance was initialized with any value other than `NoFlags`, returns `false`.

Description

[*Note:* The current instance is a subset of *target* if the current instance specifies a set of accesses to resources that is wholly contained by *target*. For example, a permission that represents read access to a file is a subset of a permission that represents read and write access to the file.

This method overrides `System.Security.CodeAccessPermission.IsSubsetOf` and is implemented to support the `System.Security.IPermission` interface.

]

Exceptions

Exception	Condition
System.ArgumentException	*target* is not `null` and is not of type `System.Security.Permissions.SecurityPermission`.

SecurityPermission.ToXml() Method

```
[ILAsm]
.method public hidebysig virtual class System.Security.SecurityElement
ToXml()

[C#]
public override SecurityElement ToXml()
```

Summary

Returns the XML encoding of the current instance.

Return Value

A `System.Security.SecurityElement` containing an XML encoding of the state of the current instance.

Description

[*Note:* For the XML encoding for this class, see the `System.Security.Permissions.SecurityPermission` class page.

This method overrides `System.Security.CodeAccessPermission.ToXml`.

]

SecurityPermission.Union(System.Security.IPermission) Method

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

[C#]
public override IPermission Union(IPermission target)
```

Summary

Returns a System.Security.Permissions.SecurityPermission object that is the union of the current instance and the specified object.

Parameters

Parameter	Description
target	A System.Security.Permissions.SecurityPermission object of the same type as the current instance to be combined with the current instance.

Return Value

A new System.Security.Permissions.SecurityPermission instance that represents the union of the current instance and *target*. If the current instance or *target* is unrestricted, returns a System.Security.Permissions.SecurityPermission instance that is unrestricted. If *target* is null, returns a copy of the current instance using the System.Security.IPermission.Copy method.

Description

[*Note:* The result of a call to System.Security.Permissions.SecurityPermission.Union is a permission that represents all of the access to security permissions represented by the current instance as well as the security permissions represented by *target*. Any demand that passes either the current instance or *target* passes their union.

This method overrides System.Security.CodeAccessPermission.Union and is implemented to support the System.Security.IPermission interface.

]

Exceptions

Exception	Condition
System.ArgumentException	*target* is not `null` and is not of type `System.Security.Permissions.SecurityPermission`.