

# System.Reflection.FieldInfo Class

```
[ILAsm]  
.class public abstract serializable FieldInfo extends  
System.Reflection.MemberInfo
```

```
[C#]  
public abstract class FieldInfo: MemberInfo
```

## 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)

## Summary

Provides access to field metadata.

## Inherits From: System.Reflection.MemberInfo

**Library:** Reflection

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

## Permissions

Permission	Description

# FieldInfo() Constructor

```
[ILAsm]  
family rtspecialname specialname instance void .ctor()
```

```
[C#]  
protected FieldInfo()
```

## Summary

Constructs a new instance of the `System.Reflection.FieldInfo` class.

# FieldInfo.GetValue(System.Object) Method

```
[ILAsm]  
.method public hidebysig virtual abstract object GetValue(object  
obj)
```

```
[C#]  
public abstract object GetValue(object obj)
```

## Summary

Obtains the value of the field that is reflected by the current instance and contained in the specified object.

## Parameters

Parameter	Description
<i>obj</i>	An object that contains the field value to be returned. If the field reflected by the current instance is static, <i>obj</i> is ignored. For non-static fields, <i>obj</i> is required to be an instance of a class that inherits or declares the field.

## Return Value

A `System.Object` that contains the value of the field reflected by the current instance.

## Behaviors

Before returning the value, the system checks to see if the user has access permission.

## Exceptions

Exception	Condition
<b>System.NotSupportedException</b>	A field is marked literal, but the field does not have one of the accepted literal types. [Note: For information regarding the accepted literal types, see Partition II of the CLI Specification.]

<b>System.FieldAccessException</b>	The field reflected by the current instance is non-public, and the caller does not have permission to access non-public members.
<b>System.ArgumentException</b>	The field reflected by the current instance is declared neither directly in <i>obj</i> nor in any class from which <i>obj</i> derives.
<b>System.Reflection.TargetException</b>	The field reflected by the current instance is non-static, and <i>obj</i> is null.

## Permissions

Permission	Description
<b>System.Security.Permissions.ReflectionPermission</b>	Requires permission to access non-public members of a type in loaded assemblies. See <code>System.Security.Permissions.ReflectionPermissionFlag.MemberAccess</code> .

# FieldInfo.SetValue(System.Object, System.Object, System.Reflection.BindingFlags, System.Reflection.Binder, System.Globalization.CultureInfo) Method

```
[ILAsm]  
.method public hidebysig virtual abstract void SetValue(object obj,  
object value, valuetype System.Reflection.BindingFlags invokeAttr,  
class System.Reflection.Binder binder, class  
System.Globalization.CultureInfo culture)  
  
[C#]  
public abstract void SetValue(object obj, object value, BindingFlags  
invokeAttr, Binder binder, CultureInfo culture)
```

## Summary

Assigns the specified value to the field that is reflected by the current instance and contained in the specified object.

## Parameters

Parameter	Description
<i>obj</i>	The object whose field value will be set. If the field is static, <i>obj</i> is ignored. For non-static fields, <i>obj</i> is required to be an instance of a class that inherits or declares the field.
<i>value</i>	An object that contains the value to assign to the field contained by <i>obj</i> .
<i>invokeAttr</i>	A System.Reflection.BindingFlags value that controls the binding process.
<i>binder</i>	A System.Reflection.Binder instance that enables the binding, coercion of argument types, and invocation of members through reflection. If <i>binder</i> is null, the default binder of the current implementation is used.
<i>culture</i>	The only defined value for this parameter is null.

## Behaviors

Before setting the value, the system verifies that the user has access permission.

## Exceptions

Exception	Condition
<b>System.ArgumentException</b>	The field reflected by the current instance is declared neither directly in <i>obj</i> nor in any class from which <i>obj</i> derives.  <i>value</i> is not assignment-compatible with the type of the field reflected by the current instance.
<b>System.FieldAccessException</b>	The field reflected by the current instance is non-public, and the caller does not have permission to access non-public members.
<b>System.Reflection.TargetException</b>	The field reflected by the current instance is non-static, and <i>obj</i> is <code>null</code> .

## Permissions

Permission	Description
<b>System.Security.Permissions.ReflectionPermission</b>	Requires permission to access non-public members of a type in loaded assemblies. See <code>System.Security.Permissions.ReflectionPermissionFlag.MemberAccess</code> .

# FieldInfo.SetValue(System.Object, System.Object) Method

```
[ILAsm]  
.method public hidebysig instance void SetValue(object obj, object  
value)  
  
[C#]  
public void SetValue(object obj, object value)
```

## Summary

Assigns the specified value to the field that is reflected by the current instance and contained in the specified object.

## Parameters

Parameter	Description
<i>obj</i>	The object whose field value will be set. If the field is static, <i>obj</i> is ignored. For non-static fields, <i>obj</i> is required to be an instance of a class that inherits or declares the field.
<i>value</i>	A <code>System.Object</code> that contains the value to assign to the field contained by <i>obj</i> .

## Description

Before setting the value, the system verifies that the user has access permission. If the user does not have access permission, a `System.FieldAccessException` is thrown.

## Exceptions

Exception	Condition
<b>System.ArgumentException</b>	The field reflected by the current instance is declared neither directly in <i>obj</i> nor in any class from which <i>obj</i> derives.  <i>value</i> is not assignment-compatible with the type of the field reflected by the current instance.
<b>System.FieldAccessException</b>	The field reflected by the current instance is non-public, and the caller does not have permission to access non-public members.

<b>System.Reflection.TargetException</b>	The field reflected by the current instance is non-static, and <i>obj</i> is null.
--	--

## Permissions

Permission	Description
<b>System.Security.Permissions.ReflectionPermission</b>	Requires permission to access non-public members of a type in loaded assemblies. See <code>System.Security.Permissions.ReflectionPermissionFlag.MemberAccess</code> .



# FieldInfo.Attributes Property

```
[ILAsm]
.property valuetype System.Reflection.FieldAttributes Attributes {
public hidebysig virtual abstract specialname valuetype
System.Reflection.FieldAttributes get_Attributes() }

[C#]
public abstract FieldAttributes Attributes { get; }
```

## Summary

Gets the attributes of the field reflected by the current instance.

## Property Value

A `System.Reflection.FieldAttributes` value that indicates the attributes of the field reflected by the current instance.

## Behaviors

This property is read-only.

## Usage

Use this property to determine the accessibility of the field reflected by the current instance. Also use this property to determine if the field reflected by the current instance can be set after it is initialized, is implemented in native code, is a literal, or has a special name.

## Example

The following example demonstrates obtaining the attributes of two fields.

```
[C#]

using System;
using System.Reflection;

class MyClass
{
    public int MyPublicInstanceField;
    private const int MyPrivateConstField = 10;
```

```

}

class FieldAttributesExample
{
    public static void Main()
    {
        Type t = (typeof(MyClass));
        string str;
        FieldInfo[] fiAry = t.GetFields( BindingFlags.Static |
            BindingFlags.Instance | BindingFlags.Public |
            BindingFlags.NonPublic | BindingFlags.DeclaredOnly );
        foreach (FieldInfo fi in fiAry)
        {
            Console.WriteLine("Field {0} is: ", fi.Name);
            str = ((fi.Attributes & FieldAttributes.Static) != 0) ?
                "Static": "Instance";
            Console.Write(str + " ");
            str = ((fi.Attributes & FieldAttributes.Public) != 0) ?
                "Public": "Not-Public";
            Console.Write(str + " ");
            str = ((fi.Attributes & FieldAttributes.Literal) != 0) ?
                "Literal": String.Empty;
            Console.WriteLine(str);
        }
    }
}

```

The output is

Field MyPublicInstanceField is:

Instance Public

Field MyPrivateConstField is:

Static Not-Public Literal

## FieldInfo.FieldType Property

```
[ILAsm]  
.property class System.Type FieldType { public hidebysig virtual  
abstract specialname class System.Type get_FieldType() }
```

```
[C#]  
public abstract Type FieldType { get; }
```

### Summary

Gets the type of the field reflected by the current instance.

### Property Value

The `System.Type` of the field reflected by the current instance.

### Description

This property is read-only.