System.Reflection.Binder Class

```
[ILASM]
.class public abstract serializable Binder extends
System.Object

[C#]
public abstract class Binder
```

Assembly Info:

- *Name:* mscorlib
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- *Version:* 1.0.x.x
- *Attributes:*
  - CLSCompliantAttribute(true)

Summary

Performs custom overload resolution and argument coercion to bind a member when reflection is used to invoke a member of a **System.Type**.

Inherits From: System.Object

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.

Description

Late binding is controlled by a customized binding interface through reflection. The **System.Reflection.Binder** class is designed to provide this functionality. **System.Reflection.Binder** objects are used in overload resolution and argument coercion for dynamic invocation of members at runtime.

Access to information obtained from reflection is controlled at two levels: untrusted code and code with **System.Security.Permissions.ReflectionPermission**.

Untrusted code is code with no special level of trust (such as code downloaded from the Internet). Such code is allowed to invoke anything that it would have been able to invoke in an early bound way.

System.Security.Permissions.ReflectionPermission controls access to metadata through reflection. If this permission is granted to

code, that code has access to all the types in its application domain, assembly, and module. It can access information about public, family, and private members of all types it has access to. Two primary capabilities are granted:

- The ability to read the metadata for family and private members of any type.
- The ability to access peer classes in the module and peer modules in the assembly.

[*Note:* The term "reflection" refers to the ability to obtain information about a **System.Object** during runtime. The primary means through which this information is accessed is via the **System.Type** of the object. Reflection allows the programmatic discovery of a type's metadata. The information included in the metadata includes details about the assembly or module in which the type is defined as well as members of the type. Reflection uses this information to provide the following primary services:

- Access to type information at runtime.
- The ability to use this type information to create instances, invoke methods, and access data members of the type.

The primary users of these services are script engines, object viewers, compilers, and object persistence formatters.

Through reflection, methods can be bound and invoked at runtime. If more than one member exists for a given member name, overload resolution determines which implementation of that method is invoked by the system. Coercion can occur when a parameter specified for a method call does not match the type specified for the parameter in the method signature. When possible, the binder converts the parameter (coerces it) to the type specified by the method signature. Coercion might not be possible depending on the types involved.

To bind to a method, field, or property, typically a list of probable candidates is obtained from the **System.Type** of a **System.Object**. That list is the passed to the appropriate method of a **System.Reflection.Binder** instance. Based on the other parameters passed to that method, typically (although not necessarily) one of the members of the list is chosen, and an object that reflects that member is returned.

The system supplies a default binder that provides default binding rules. Because binding rules vary among programming languages, it is recommended that each programming language provide a custom implementation of **System.Reflection.Binder**.]

Binder() Constructor

```
[ILASM]
family specialname instance void .ctor()

[C#]
protected Binder()
```

Summary

Constructs a new instance of the **System.Reflection.Binder** class.

Binder.BindToField(System.Reflection.BindingFlags, System.Reflection.FieldInfo[], System.Object, System.Globalization.CultureInfo) Method

```
[ILASM]
.method public hidebysig virtual abstract class
System.Reflection.FieldInfo BindToField(valuetype
System.Reflection.BindingFlags bindingAttr, class
System.Reflection.FieldInfo[] match, object value, class
System.Globalization.CultureInfo culture)

[C#]
public abstract FieldInfo BindToField(BindingFlags
bindingAttr, FieldInfo[] match, object value, CultureInfo
culture)
```

Summary

Selects a field from the specified set of fields, based on the specified criteria.

Parameters

Parameter	Description
bindingAttr	A bitwise combination of **System.Reflection.BindingFlags** values that control the binding process. For requirements, see the Behaviors section.
match	An array of **System.Reflection.FieldInfo** objects whose elements represent the set of fields that reflection has determined to be a possible match, typically because the fields have the correct member name.
value	An object of a type that is assignment-compatible with the type of the field being searched for. [*Note:* For example, if *value* is an instance of a class, the type of that instance can be assigned to the type of the field returned by this method. Fields in *match* that cannot be assigned to this value are eliminated from the search.]
culture	The only defined value for this parameter is **null**.

Return Value

A **System.Reflection.FieldInfo** instance that reflects the field that matches the specified criteria. It is not required that this instance be contained in *match*. If a suitable field is not found, returns **null**.

Behaviors

For the *bindingAttr* parameter, the caller is required to specify either **System.Reflection.BindingFlags.Public** or **System.Reflection.BindingFlags.NonPublic**, and either **System.Reflection.BindingFlags.Instance** or **System.Reflection.BindingFlags.Static**. If at least one value from each pair is not specified, this method is required to return **null**.

Binder.BindToMethod(System.Reflection.BindingFlags, System.Reflection.MethodBase[], System.Object[]&, System.Reflection.ParameterModifier[], System.Globalization.CultureInfo, System.String[], System.Object&) Method

```
[ILASM]
.method public hidebysig virtual abstract class
System.Reflection.MethodBase BindToMethod(valuetype
System.Reflection.BindingFlags bindingAttr, class
System.Reflection.MethodBase[] match, class
System.Object[]& args, class
System.Reflection.ParameterModifier[] modifiers, class
System.Globalization.CultureInfo culture, class
System.String[] names, class System.Object& state)

[C#]
public abstract MethodBase BindToMethod(BindingFlags
bindingAttr, MethodBase[] match, ref object[] args,
ParameterModifier[] modifiers, CultureInfo culture,
string[] names, ref object state)
```

Summary

Selects a method based on the specified criteria.

Parameters

Parameter	Description
bindingAttr	A bitwise combination of **System.Reflection.BindingFlags** values that control the binding process. For requirements, see the Behaviors section.
match	An array of **System.Reflection.MethodBase** objects that represent the set of methods that Reflection has determined to be a possible match, typically because they have the correct member name.
args	An array of objects that represent the parameters passed in the method invocation. The types, values, and order of the elements of this array might be changed by this method to match the signature of the selected method.
modifiers	The only defined value for this parameter is **null**.

culture	The only defined value for this parameter is **null**.
names	A **System.String** array containing the names of methods to be searched.
state	A binder-provided **System.Object** that keeps track of parameter reordering. The *state* object is totally defined by the implementer of the **System.Reflection.Binder** class. This object is **null** if the binder does not reorder the argument array of the bound method.

Return Value

A **System.Reflection.MethodBase** instance that reflects the method that matches to the specified criteria. It is not required that this instance be contained in *match*. If a suitable method is not found, returns **null**.

Description

If *state* is not **null**, the system invokes **System.Reflection.Binder.ReorderArgumentArray** after this method returns. [*Note:* This allows a caller to map the argument array of a method back to the original form if the order has been altered by **System.Reflection.Binder.BindToMethod**. This is useful if **ByRef** arguments are in the argument array, because the caller can retrieve those arguments in their original order on return from this method. When arguments are passed by name (i.e., using named arguments), the binder reorders the argument array and that is what the caller sees. This method insures that the original order of the arguments is restored.]

Behaviors

For the *bindingAttr* parameter, the caller is required to specify either **System.Reflection.BindingFlags.Public** or **System.Reflection.BindingFlags.NonPublic**, and either **System.Reflection.BindingFlags.Instance** or **System.Reflection.BindingFlags.Static**. If at least one value from each pair is not specified, this method is required to return **null**.

The **System.Reflection.Binder.BindToMethod** method is permitted to change the order of the argument array of a method call only if the binder returns, via the *state* parameter, a non-null opaque object that records the original order of the argument array. If, on return from **System.Reflection.Binder.BindToMethod**, *state* is not **null**, the system calls **System.Reflection.Binder.ReorderArgumentArray**.

Binder.ChangeType(System.Object, System.Type, System.Globalization.CultureInfo) Method

```
[ILASM]
.method public hidebysig virtual abstract object
ChangeType(object value, class System.Type type, class
System.Globalization.CultureInfo culture)

[C#]
public abstract object ChangeType(object value, Type type,
CultureInfo culture)
```

Summary

Converts the type of the specified object to the specified type.

Parameters

Parameter	Description
value	The object to be converted to a new **System.Type**.
type	The **System.Type** to which *value* is converted.
culture	The only defined value for this parameter is **null**.

Return Value

A new object of the type specified by *type.* The contents of this object are equal to those of *value*.

Behaviors

As described above.

How and When to Override

Implement this method to change the type of a member of a parameter array. Typically, it is recommended that implementations of this method perform only widening conversions.

Usage

This method is used to change the type of a element in a parameter array to match the type required by the signature of a bound method.

Binder.ReorderArgumentArray(System.Object[]&, System.Object) Method

```
[ILASM]
.method public hidebysig virtual abstract void
ReorderArgumentArray(class System.Object[]& args, object
state)

[C#]
public abstract void ReorderArgumentArray(ref object[]
args, object state)
```

Summary

Restores the specified set of parameters to their original order after a call to **System.Reflection.Binder.BindToMethod**.

Parameters

Parameter	Description
args	An array of objects whose elements represent the parameters passed to the bound method in their original order.
state	A binder-provided opaque object that keeps track of parameter reordering. This object is the same object that was passed as the *state* parameter in the invocation of **System.Reflection.Binder.BindToMethod** that caused **System.Reflection.Binder.ReorderArgumentArray** to be called.

Description

[*Note:* When a method call is bound to a method through reflection using **System.Reflection.Binder.BindToMethod**, the order, value, and type of the parameters in the original method call may be changed to match the signature of the bound method. The binder creates *state* as an opaque object that records the original order of the argument array. If, on return from **System.Reflection.Binder.BindToMethod**, *state* is not **null**, the system calls **System.Reflection.Binder.ReorderArgumentArray**. This allows a caller to map the argument array of a method back to the original form if the order had been altered by **System.Reflection.Binder.BindToMethod**. This is useful if **ByRef** arguments are in the argument array, because the caller can retrieve those arguments in their original order on return from this method. When arguments are passed by name (i.e., using named arguments), the binder reorders the argument array and that is what the caller sees. This method insures that the original order of the arguments is restored.]

Behaviors

state is required to be a non-null **System.Object** that tracks the original ordering of *args* if *args* is reordered by a call to **System.Reflection.Binder.BindToMethod**. This method is required to restore the elements of *args* to their original order, value, and **System.Type**

How and When to Override

Implement this method to insure that the parameters contained in *args* are returned to their original order, **System.Type** and value, after being used by a bound method.

Usage

Use this method to insure that the parameters contained in *args* are returned to their original order, **System.Type** and value, after being used by a bound method.

Binder.SelectMethod(System.Reflection.BindingFlags, System.Reflection.MethodBase[], System.Type[], System.Reflection.ParameterModifier[]) Method

```
[ILASM]
.method public hidebysig virtual abstract class
System.Reflection.MethodBase SelectMethod(valuetype
System.Reflection.BindingFlags bindingAttr, class
System.Reflection.MethodBase[] match, class System.Type[]
types, class System.Reflection.ParameterModifier[]
modifiers)

[C#]
public abstract MethodBase SelectMethod(BindingFlags
bindingAttr, MethodBase[] match, Type[] types,
ParameterModifier[] modifiers)
```

Summary

Selects a method from the specified set of methods, based on the argument type.

Parameters

Parameter	Description
bindingAttr	A bitwise combination of **System.Reflection.BindingFlags** values that control the binding process. For requirements, see the Behaviors section.
match	An array of **System.Reflection.MethodBase** objects that represent the set of methods that Reflection has determined to be a possible match, typically because they have the correct member name.
types	An array of **System.Type** objects that represent the values used to locate a matching method.
modifiers	The only defined value for this parameter is **null**.

Return Value

A **System.Reflection.MethodBase** instance that reflects the method that is matched to the specified criteria. It is not required that this

instance be contained in *match*. If a suitable method is not found, returns **null**.

Behaviors

For the *bindingAttr* parameter, the caller is required to specify either **System.Reflection.BindingFlags.Public** or **System.Reflection.BindingFlags.NonPublic**, and either **System.Reflection.BindingFlags.Instance** or **System.Reflection.BindingFlags.Static**. If at least one value from each pair is not specified, this method is required to return **null**.

Binder.SelectProperty(System.Reflection.BindingFlags, System.Reflection.PropertyInfo[], System.Type, System.Type[], System.Reflection.ParameterModifier[]) Method

```
[ILASM]
.method public hidebysig virtual abstract class
System.Reflection.PropertyInfo SelectProperty(valuetype
System.Reflection.BindingFlags bindingAttr, class
System.Reflection.PropertyInfo[] match, class System.Type
returnType, class System.Type[] indexes, class
System.Reflection.ParameterModifier[] modifiers)

[C#]
public abstract PropertyInfo SelectProperty(BindingFlags
bindingAttr, PropertyInfo[] match, Type returnType, Type[]
indexes, ParameterModifier[] modifiers)
```

Summary

Selects a property from the specified set of properties, based on the specified criteria.

Parameters

Parameter	Description
bindingAttr	A bitwise combination of **System.Reflection.BindingFlags** values that control the binding process. For requirements, see the Behaviors section.
match	An array of **System.Reflection.PropertyInfo** objects that represent the set of properties that Reflection has determined to be a possible match, typically because they have the correct member name.
returnType	The **System.Type** of the property being searched for.
indexes	An array of **System.Type** objects that represent the index types of the property being searched for. [*Note:* Use this parameter for index properties such as the indexer for a class.]
modifiers	The only defined value for this parameter is **null**.

Return Value

A **System.Reflection.PropertyInfo** instance that reflects the property that matches the specified criteria. It is not required that this instance be contained in *match*. If a suitable property is not found, returns **null**.

Behaviors

For the *bindingAttr* parameter, the caller is required to specify either **System.Reflection.BindingFlags.Public** or **System.Reflection.BindingFlags.NonPublic**, and either **System.Reflection.BindingFlags.Instance** or **System.Reflection.BindingFlags.Static**. If at least one value from each pair is not specified, this method is required to return **null**.