

1 System.Enum Class

```
2 [ILAsm]  
3 .class public abstract serializable Enum extends System.ValueType  
4 implements System.IComparable, System.IFormattable  
5 [C#]  
6 public abstract class Enum: ValueType, IComparable, IFormattable
```

7 Assembly Info:

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

13 Implements:

- 14 • **System.IComparable**
- 15 • **System.IFormattable**

16 Summary

17 Provides support for all enumeration types. Serves as the base class for all enumeration
18 types.

19 Inherits From: System.ValueType

20

21 **Library:** BCL

22

23 Description

24 A `System.Enum` is a distinct type with named constant members. Each enumeration type
25 has a corresponding integral type called the *underlying type* of the enumeration type.
26 This underlying type is required to be a system-supplied integer type that is large
27 enough to represent all values defined in the enumeration; the field that holds the
28 underlying type must be called `value__`. A `System.Enum` declaration is allowed to
29 explicitly declare any integral type other than `System.Char` as an underlying type. This
30 includes `System.Byte`, `System.SByte`, `System.Int16`, `System.Int32`, `System.Int64`,
31 `System.UInt16`, `System.UInt32`, and `System.UInt64`. A `System.Enum` declaration that
32 does not explicitly declare an underlying type has an underlying type of `System.Int32`.

33

34 `System.Enum` derives from `System.ValueType` but is not a value type. Programming
35 languages typically provide syntax to declare sets of a specified enumeration type
36 consisting of named constants and their values.

37

38 It is possible to treat instances of a `System.Enum` as bit fields containing multiple values.
39 For more information, see `System.FlagsAttribute`.

40

1 [Note: `System.Enum` provides methods to compare instances of enumeration types,
2 convert the value of an instance to its `System.String` representation, convert the
3 `System.String` representation of a number to an instance of the enumeration type, and
4 create an instance of a specified enumeration and value.]

5
6

7

1 Enum.CompareTo(System.Object) Method

```
2 [ILAsm]  
3 .method public final hidebysig virtual int32 CompareTo(object target)  
4 [C#]  
5 public int CompareTo(object target)
```

6 Summary

7 Returns the sort order of the current instance compared to the specified `System.Object`.

8 Parameters

Parameter	Description
<i>target</i>	An object to compare the current instance to.

9

10 Return Value

11 The return value is a negative number, zero, or a positive number reflecting the sort
12 order of the current instance as compared to *target*. For non-zero return values, the
13 exact value returned by this method is unspecified. The following table defines the
14 return value:

Return Value	Description
A negative integer	The value of the current instance is less than the value of <i>target</i> .
Zero	The value of the current instance is equal to the value of <i>target</i> .
Any positive integer	The value of the current instance is greater than the value of <i>target</i> , or <i>target</i> is null.

15

16 Description

17 [Note: This method is implemented to support the `System.IComparable` interface.]
18
19

20 Exceptions

Exception	Condition
System.ArgumentException	<i>target</i> and the current instance are not of the same enumeration type.

1

2

1 Enum.Equals(System.Object) Method

```
2 [ILAsm]  
3 .method public hidebysig virtual bool Equals(object obj)  
4 [C#]  
5 public override bool Equals(object obj)
```

6 Summary

7 Determines whether the current instance and the specified `System.Object` represent the
8 same type and value.

9 Parameters

Parameter	Description
<i>obj</i>	An object to compare the current instance to.

10

11 Return Value

12 `true` if *obj* is of the same enumeration type and represents the same value as the
13 current instance; otherwise, `false`.

14 Description

15 [*Note:* This method overrides `System.Object.Equals`.]
16
17

18

1 Enum.Format(System.Type, System.Object, 2 System.String) Method

```
3 [ILAsm]  
4 .method public hidebysig static string Format(class System.Type enumType,  
5 object value, string format)  
  
6 [C#]  
7 public static string Format(Type enumType, object value, string format)
```

8 Summary

9 Converts the specified element of the specified enumeration type to its System.String
10 representation using the specified format.

11 Parameters

Parameter	Description
<i>enumType</i>	A System.Type that specifies the type of the enumeration of which <i>value</i> is a member.
<i>value</i>	The enumeration element to be converted.
<i>format</i>	A System.String that specifies the output format to use.

12 13 Return Value

14 The System.String representation of the value of the enumeration element.

15 Description

16 For cross-platform portability, the only valid values for *format* are:

Format	Description
"G" or "g"	If <i>value</i> is equal to a defined value of <i>enumType</i> , returns the element name defined for <i>value</i> . If the System.FlagsAttribute attribute is set on the System.Enum declaration and <i>value</i> is a built-in integer type and is equal to a summation of enumeration elements, the return value contains the element names in an unspecified order, separated by commas (e.g. "Red, Yellow"). Otherwise, <i>value</i> is returned in decimal format.
"X" or "x"	Returns <i>value</i> in hexadecimal format, without a leading 0x. The value is padded

"x"	with leading zeroes to ensure the returned value is at least eight digits in length.
"F" or "f"	Behaves identically to "G", except the <code>FlagsAttribute</code> is not required to be present on the <code>System.Enum</code> declaration.
"D" or "d"	Returns <i>value</i> in decimal format with no leading zeroes.

1

2 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> , <i>value</i> or <i>format</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <code>System.Enum</code> . -or- <i>value</i> is neither of type <i>enumType</i> nor does it have the same underlying type as <i>enumType</i> .
System.FormatException	<i>format</i> contains an invalid value.

3

4 Example

5 The following example demonstrates formatting enumeration values.

6

7 [C#]

```

8 using System;
9 public enum Signs {
10     Stop = 1,
11     Yield = 2,
12     Merge = 4
13 };
14 [Flags]
15 public enum Lights {
16     Red = 1,
17     Yellow = 2,
18     Green = 4
19 };
20 public class EnumCompTo {
21     public static void Main() {
22         Console.WriteLine(Signs.Format(typeof(Signs), Signs.Merge, "d"));
23         Console.WriteLine(Signs.Format(typeof(Signs), 7, "g"));

```

```
1     Console.WriteLine(Lights.Format(typeof(Lights), Lights.Yellow, "x"));
2     Console.WriteLine(Lights.Format(typeof(Lights), 7, "g"));
3 }
4 }
5
6 The output is
7
8 4
9
10
11 7
12
13
14 00000002
15
16
17 Red, Yellow, Green
18
19
```

1 Enum.GetHashCode() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual int32 GetHashCode()  
4 [C#]  
5 public override int GetHashCode()
```

6 Summary

7 Generates a hash code for the current instance.

8 Return Value

9 A `System.Int32` containing a hash code for the current instance.

10 Description

11 The algorithm used to generate the hash code is unspecified.

12

13 [*Note:* This method overrides `System.Object.GetHashCode()`.]

14

15

16

1 Enum.GetName(System.Type, System.Object) 2 Method

```
3 [ILAsm]  
4 .method public hidebysig static string GetName(class System.Type enumType,  
5 object value)  
  
6 [C#]  
7 public static string GetName(Type enumType, object value)
```

8 Summary

9 Retrieves the name of the constant of the specified enumeration type that has the
10 specified value.

11 Parameters

Parameter	Description
<i>enumType</i>	A <code>System.Type</code> that specifies the type of the enumeration.
<i>value</i>	A <code>System.Object</code> that contains the integral value or the name of an enumeration constant.

12

13 Return Value

14 A `System.String` containing the name of the enumerated constant in *enumType* whose
15 value is *value*, or a null reference if no such constant is found. If multiple constants have
16 the same value, as to which name is returned for that value, is unspecified.

17 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> or <i>value</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <code>System.Type</code> that describes a <code>System.Enum</code> . -or- <i>value</i> is neither of type <i>enumType</i> nor does it have the same underlying type as <i>enumType</i> .

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

Example

The following example demonstrates the System.Enum.GetName method.

[C#]

```
using System;
public enum Signs {
    Stop = 1,
    Yield = 2,
    Merge = 4
};
public class EnumCompTo {
    public static void Main() {
        Console.Write( "The name of the constant with the value 4 is: " );
        Console.WriteLine( "{0}.", Signs.GetName(typeof(Signs), 4));
        Console.Write( "The name of the constant with the value Stop is: " );
        Console.WriteLine( "{0}.", Signs.GetName(typeof(Signs), Signs.Stop ));
    }
}
```

The output is

The name of the constant with the value 4 is: Merge.

The name of the constant with the value Stop is: Stop.

1 Enum.GetNames(System.Type) Method

```
2 [ILAsm]  
3 .method public hidebysig static string[] GetNames(class System.Type  
4 enumType)  
  
5 [C#]  
6 public static string[] GetNames(Type enumType)
```

7 Summary

8 Returns a zero-based, one-dimensional `System.String` array that contains the names of
9 the constants of the specified enumeration type.

10 Parameters

Parameter	Description
<code>enumType</code>	A <code>System.Type</code> that specifies the type of an enumeration.

11 Return Value

13 A `System.String` vector of the names of the constants in `enumType`. The elements of
14 the vector are sorted by the values of the enumerated constants. If multiple constants
15 have the same value, the order in which their names appear in the vector, relative to
16 each other, is unspecified.

17 Exceptions

Exception	Condition
System.ArgumentNullException	<code>enumType</code> is a null reference.
System.ArgumentException	<code>enumType</code> is not a <code>System.Type</code> that describes a <code>System.Enum</code> .

18 Example

20 The following example demonstrates the `System.Enum.GetNames` method.

```
21 [C#]  
22  
23 using System;  
24  
25 public enum Colors {  
26     Red,
```

```
1     White,  
2     Blue  
3 };  
4  
5 public class enumGetNames {  
6  
7     public static void Main() {  
8         int i = 0;  
9         String[] strAry = Colors.GetNames( typeof(Colors) );  
10        foreach (String str in strAry) {  
11            Console.Write("The value indexed '{0}' ", i++ );  
12            Console.WriteLine("is {0}.", str);  
13        }  
14    }  
15 }
```

16 The output is

```
17  
18 The value indexed '0' is Red.  
19  
20  
21 The value indexed '1' is White.  
22  
23  
24 The value indexed '2' is Blue.  
25
```

26

1 Enum.GetUnderlyingType(System.Type)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig static class System.Type GetUnderlyingType(class  
5 System.Type enumType)  
  
6 [C#]  
7 public static Type GetUnderlyingType(Type enumType)
```

8 Summary

9 Returns the underlying type of the specified enumeration type.

10 Parameters

Parameter	Description
<i>enumType</i>	The <code>System.Type</code> of an enumeration.

11 Return Value

13 A `System.Type` instance that describes the underlying type of *enumType*.

14 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not an enumeration type.

15 Example

17 The following example demonstrates the `System.Enum.GetUnderlyingType` method.

```
18 [C#]  
19  
20 using System;  
21 public enum Colors {  
22     Red,  
23     White,  
24     Blue  
25 }  
26 public class EnumUnderlyingTypeTest {  
27     public static void Main() {
```

```
1     Type t = Colors.GetUnderlyingType( typeof(Colors) );
2     Console.WriteLine("The underlying type of Colors is {0}.",
3 t.ToString());
4 }
5 }
6 The output is
7
8 The underlying type of Colors is System.Int32.
9
10
```

1 Enum.GetValues(System.Type) Method

```
2 [ILAsm]  
3 .method public hidebysig static class System.Array GetValues(class  
4 System.Type enumType)  
  
5 [C#]  
6 public static Array GetValues(Type enumType)
```

7 Summary

8 Returns a zero-based, one-dimensional array of the values of the constants of the
9 specified enumeration type.

10 Parameters

Parameter	Description
<i>enumType</i>	The <code>System.Type</code> of an enumeration.

11 Return Value

13 A vector of the enumeration type specified by *enumType* containing the values of the
14 constants in *enumType*. The elements of the array are sorted by the values of the
15 enumeration constants. If multiple constants have the same value, the value of each is
16 included in the array.

17 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not an enumeration type.

18 Example

20 The following example demonstrates the `System.Enum.GetValues` method.

```
21 [C#]  
22  
23 using System;  
24 public enum Colors {  
25     Red = 1,  
26     White = 2,  
27     Blue = 4  
28 }
```

```
1 public class enumGetValues {
2     public static void Main() {
3         Array valueAry = Enum.GetValues(typeof(Colors));
4         foreach (int i in valueAry) {
5             Console.WriteLine("The value of element {0} is {1}",
6                 Enum.Format(typeof(Colors), i, "g"),i);
7         }
8     }
9 }
```

10 The output is

```
11
12 The value of element Red is 1.
13
14
15 The value of element White is 2.
16
17
18 The value of element Blue is 4.
19
```

20

1 Enum.IsDefined(System.Type, 2 System.Object) Method

```
3 [ILAsm]  
4 .method public hidebysig static bool IsDefined(class System.Type enumType,  
5 object value)  
  
6 [C#]  
7 public static bool IsDefined(Type enumType, object value)
```

8 Summary

9 Returns a `System.Boolean` indicating whether a constant with the specified value exists
10 in the specified enumeration type.

11 Parameters

Parameter	Description
<i>enumType</i>	A <code>System.Type</code> that describes an enumeration.
<i>value</i>	The constant or value being searched for in <i>enumType</i> .

12 13 Return Value

14 `true` if a constant in the enumeration described by *enumType* has a value equal to
15 *value*; otherwise, `false`.

16 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> or <i>value</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not an enumeration type. -or- The type of <i>value</i> is not an <i>enumType</i> or an underlying type of <i>enumType</i> .

17 18 Example

1 The following example demonstrates the System.Enum.IsDefined method.

2

3 [C#]

```
4 using System;
5 public enum Colors {
6     Red = 1,
7     White = 2,
8     Blue = 4
9 }
10 public class enumIsDefined {
11     public static void Main() {
12         Console.Write("It is {0} ", Enum.IsDefined(typeof(Colors), 1));
13         Console.WriteLine("that '1' is defined in 'Colors'.");
14         Console.Write("It is {0} ", Enum.IsDefined(typeof(Colors), 3));
15         Console.WriteLine("that '3' is defined in 'Colors'.");
16     }
17 }
```

18 The output is

19

20 It is True that '1' is defined in 'Colors'.

21

22

23 It is False that '3' is defined in 'Colors'.

24

25

1 Enum.Parse(System.Type, System.String)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig static object Parse(class System.Type enumType,  
5 string value)  
  
6 [C#]  
7 public static object Parse(Type enumType, string value)
```

8 Summary

9 Converts the specified `System.String` representation of one or more enumerated
10 constants of a specified enumeration type to an equivalent enumerated object.

11 Parameters

Parameter	Description
<i>enumType</i>	The <code>System.Type</code> of an enumeration.
<i>value</i>	A <code>System.String</code> containing one or more names or a single numeric value to convert. If the string contains more than one name, each name is required to be separated by a comma (','),. The names are parsed in a case-sensitive manner. The names or number can be surrounded by any amount of white space.

12

13 Return Value

14 A `System.Object` of type *enumType* whose values are represented by *value*.

15 Description

16 This version of `System.Enum.Parse` is equivalent to `System.Enum.Parse (enumType,`
17 `value, false)`.

18 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> or <i>value</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <code>System.Type</code> that describes a <code>System.Enum</code> .

	<p>-or-</p> <p><i>value</i> is either equal to <code>System.String.Empty</code> or contains only white space.</p> <p>-or-</p> <p><i>value</i> represents one or more names, and at least one name represented by <i>value</i> is not of type <i>enumType</i>.</p>
--	---

1

2 **Example**

3 The following example demonstrates the `System.Enum.Parse` method.

4

5 [C#]

```

6 using System;
7 public enum Colors {
8     Red = 1,
9     Blue = 2
10 }
11 public class enumTest {
12     public static void Main() {
13         object o = Enum.Parse( typeof (Colors), "Red, Blue");
14         Type oType = o.GetType();
15         Console.WriteLine("The type of the object returned is
16 {0}",oType.ToString());
17         Array values = Enum.GetValues(oType);
18         foreach (Colors c in values) {
19             Console.WriteLine(Enum.Format(oType,c,"D"));
20         }
21     }
22 }

```

23 The output is

24

25 The type of the object returned is Colors

26

27

28 1

29

30

31 2

32

33

1 Enum.Parse(System.Type, System.String, 2 System.Boolean) Method

```
3 [ILAsm]  
4 .method public hidebysig static object Parse(class System.Type enumType,  
5 string value, bool ignoreCase)  
  
6 [C#]  
7 public static object Parse(Type enumType, string value, bool ignoreCase)
```

8 Summary

9 Converts the specified `System.String` representation of one or more enumerated
10 constants of a specified enumeration type to an equivalent enumerated object. This
11 method behaves in a case-sensitive or insensitive manner according to the specified
12 `System.Boolean`.

13 Parameters

Parameter	Description
<i>enumType</i>	The <code>System.Type</code> of an enumeration.
<i>value</i>	A <code>System.String</code> containing one or more names or a single numeric value to convert. If the string contains more than one name, each name is required to be separated by a comma (','),. The names or number can be surrounded by any amount of white space.
<i>ignoreCase</i>	A <code>System.Boolean</code> value. Specify <code>true</code> to have names in <i>value</i> parsed in a case-insensitive manner. Specify <code>false</code> to have names parsed in a case-sensitive manner.

14 15 Return Value

16 A `System.Object` of type *enumType* whose values are represented by *value*.

17 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> or <i>value</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <code>System.Type</code> that describes a <code>System.Enum</code> .

	<p>-or-</p> <p><i>value</i> is either equal to <code>System.String.Empty</code> or contains only white space.</p> <p>-or-</p> <p><i>value</i> represents one or more names, and at least one name represented by <i>value</i> is not of type <i>enumType</i>.</p>
--	---

1

2 **Example**

3 For an example that demonstrates parsing strings containing enumeration values, see
4 `System.Enum.Parse(System.Type, System.String)`.

5

1 Enum.ToObject(System.Type, System.Object) 2 Method

```
3 [ILAsm]  
4 .method public hidebysig static object ToObject(class System.Type  
5 enumType, object value)  
  
6 [C#]  
7 public static object ToObject(Type enumType, object value)
```

8 Summary

9 Returns an instance of the specified enumeration type set to the specified value.

10 Parameters

Parameter	Description
<i>enumType</i>	The <i>System.Type</i> of the enumeration.
<i>value</i>	The value to set.

11 12 Return Value

13 An enumeration object of type *enumType* whose value is *value*.

14 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <i>System.Type</i> that describes a <i>System.Enum</i> .

15
16

1 Enum.ToObject(System.Type, System.SByte) 2 Method

```
3 [ILAsm]  
4 .method public hidebysig static object ToObject(class System.Type  
5 enumType, int8 value)  
  
6 [C#]  
7 public static object ToObject(Type enumType, sbyte value)
```

8 Summary

9 Returns an instance of the specified enumeration type set to the specified `System.SByte`
10 value.

11 Type Attributes:

- 12 • CLSCompliantAttribute(false)

13 Parameters

Parameter	Description
<i>enumType</i>	The <code>System.Type</code> of an enumeration.
<i>value</i>	The <code>System.SByte</code> value to set.

14 15 Return Value

16 An enumeration object of type *enumType* whose value is *value*.

17 Description

18 This member is not CLS-compliant. For a CLS-compliant alternative, use
19 `System.Enum.ToObject(System.Type, System.Int16)`.

20 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <code>System.Type</code> that describes a <code>System.Enum</code> .

1

2

1 Enum.ToObject(System.Type, System.Int16)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig static object ToObject(class System.Type  
5 enumType, int16 value)  
  
6 [C#]  
7 public static object ToObject(Type enumType, short value)
```

8 Summary

9 Returns an instance of the specified enumeration type set to the specified `System.Int16`
10 value.

11 Parameters

Parameter	Description
<i>enumType</i>	The <code>System.Type</code> of an enumeration.
<i>value</i>	The <code>System.Int16</code> value to set.

12

13 Return Value

14 An enumeration object of type *enumType* whose value is *value*.

15 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <code>System.Type</code> that describes a <code>System.Enum</code> .

16

17

1 Enum.ToObject(System.Type, 2 System.UInt64) Method

```
3 [ILAsm]  
4 .method public hidebysig static object ToObject(class System.Type  
5 enumType, unsigned int64 value)  
  
6 [C#]  
7 public static object ToObject(Type enumType, ulong value)
```

8 Summary

9 Returns an instance of the specified enumeration type set to the specified
10 System.UInt64 value.

11 Type Attributes:

- 12 • CLSCompliantAttribute(false)

13 Parameters

Parameter	Description
<i>enumType</i>	The System.Type of an enumeration.
<i>value</i>	The System.UInt64 value to set.

14 15 Return Value

16 An enumeration object of type *enumType* whose value is *value*.

17 Description

18 This member is not CLS-compliant. For a CLS-compliant alternative, use
19 System.Enum.ToObject(System.Type, System.Int64).

20 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a System.Type that describes a System.Enum.

1

2

1 Enum.ToObject(System.Type, System.Int64)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig static object ToObject(class System.Type  
5 enumType, int64 value)  
  
6 [C#]  
7 public static object ToObject(Type enumType, long value)
```

8 Summary

9 Returns an instance of the specified enumeration type set to the specified `System.Int64`
10 value.

11 Parameters

Parameter	Description
<i>enumType</i>	The <code>System.Type</code> of an enumeration.
<i>value</i>	The <code>System.Int64</code> value to set.

12

13 Return Value

14 An enumeration object of type *enumType* whose value is *value*.

15 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <code>System.Type</code> that describes a <code>System.Enum</code> .

16

17

1 Enum.ToObject(System.Type, 2 System.UInt32) Method

```
3 [ILAsm]  
4 .method public hidebysig static object ToObject(class System.Type  
5 enumType, unsigned int32 value)  
  
6 [C#]  
7 public static object ToObject(Type enumType, uint value)
```

8 Summary

9 Returns an instance of the specified enumeration type set to the specified
10 System.UInt32 value.

11 Type Attributes:

- 12 • CLSCompliantAttribute(false)

13 Parameters

Parameter	Description
<i>enumType</i>	The System.Type of an enumeration.
<i>value</i>	The System.UInt32 value to set.

14 15 Return Value

16 An enumeration object of type *enumType* whose value is *value*.

17 Description

18 This member is not CLS-compliant. For a CLS-compliant alternative, use
19 System.Enum.ToObject(System.Type, System.Int64).

20 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a System.Type that describes a System.Enum.

1

2

1 Enum.ToObject(System.Type, 2 System.UInt16) Method

```
3 [ILAsm]  
4 .method public hidebysig static object ToObject(class System.Type  
5 enumType, unsigned int16 value)  
  
6 [C#]  
7 public static object ToObject(Type enumType, ushort value)
```

8 Summary

9 Returns an instance of the specified enumeration type set to the specified
10 System.UInt16 value.

11 Type Attributes:

- 12 • CLSCompliantAttribute(false)

13 Parameters

Parameter	Description
<i>enumType</i>	The System.Type of an enumeration.
<i>value</i>	The System.UInt16 value to set.

14 15 Return Value

16 An enumeration object of type *enumType* whose value is *value*.

17 Description

18 This member is not CLS-compliant. For a CLS-compliant alternative, use
19 System.Enum.ToObject(System.Type, System.Int32).

20 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a System.Type that describes a System.Enum.

1

2

1 Enum.ToObject(System.Type, System.Byte)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig static object ToObject(class System.Type  
5 enumType, unsigned int8 value)  
  
6 [C#]  
7 public static object ToObject(Type enumType, byte value)
```

8 Summary

9 Returns an instance of the specified enumeration type set to the specified `System.Byte`
10 value.

11 Parameters

Parameter	Description
<i>enumType</i>	The <code>System.Type</code> of an enumeration.
<i>value</i>	The <code>System.Byte</code> value to set.

12

13 Return Value

14 An enumeration object of type *enumType* whose value is *value*.

15 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <code>System.Type</code> that describes a <code>System.Enum</code> .

16

17

1 Enum.ToObject(System.Type, System.Int32)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig static object ToObject(class System.Type  
5 enumType, int32 value)  
  
6 [C#]  
7 public static object ToObject(Type enumType, int value)
```

8 Summary

9 Returns an instance of the specified enumeration type set to the specified `System.Int32`
10 value.

11 Parameters

Parameter	Description
<i>enumType</i>	The <code>System.Type</code> of an enumeration.
<i>value</i>	The <code>System.Int32</code> value to set.

12

13 Return Value

14 An enumeration object of type *enumType* whose value is *value*.

15 Exceptions

Exception	Condition
System.ArgumentNullException	<i>enumType</i> is a null reference.
System.ArgumentException	<i>enumType</i> is not a <code>System.Type</code> that describes a <code>System.Enum</code> .

16

17

1 Enum.ToString(System.String) Method

```
2 [ILAsm]  
3 .method public hidebysig instance string ToString(string format)  
4 [C#]  
5 public string ToString(string format)
```

6 Summary

7 Converts the value of the current instance to its equivalent `System.String`
8 representation using the specified format.

9 Parameters

Parameter	Description
<i>format</i>	A <code>System.String</code> that specifies the format to use when converting the current instance to a <code>System.String</code> . Specify one of the following values in upper or lowercase: "G", "D", "X", or "F".

10 11 Return Value

12 The `System.String` representation of the value of the current instance as specified by
13 *format*.

14 Description

15 If *format* is a null reference or an empty string, the return value is formatted using the
16 general format specifier ("G").

17
18 [Note: For details on the format specifiers used with an enumeration object, see
19 `System.Enum.Format`.]
20
21

22 Exceptions

Exception	Condition
<code>System.FormatException</code>	<i>format</i> contains an invalid value.

23
24

1 Enum.ToString() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual string ToString()  
4 [C#]  
5 public override string ToString()
```

6 Summary

7 Converts the name of the value of the current instance to its equivalent `System.String`
8 representation.

9 Return Value

10 The `System.String` representation of the named constant specified by the current
11 instance.

12 Description

13 This version of `System.Enum.ToString` is equivalent to `System.Enum.ToString ("G",`
14 `null)`.

15
16 This method returns the same value as `System.Enum.Format` with the "g" or "G" format
17 option.

18
19 An instance of an enumeration is set to a named constant value. This method returns
20 the name of the constant an instance is set to, not the value itself.

21

1 Enum.ToString(System.String, 2 System.IFormatProvider) Method

```
3 [ILAsm]  
4 .method public final hidebysig virtual string ToString(string format,  
5 class System.IFormatProvider provider)  
  
6 [C#]  
7 public string ToString(string format, IFormatProvider provider)
```

8 Summary

9 Converts the numeric value of the current instance to its equivalent System.String
10 representation using the specified format.

11 Parameters

Parameter	Description
<i>format</i>	A System.String that specifies the format to use when converting the current instance to a System.String. Specify one of the following values in upper or lowercase: "G", "D", "X", or "F".
<i>provider</i>	An object that implements the System.IFormatProvider interface or a null reference. The System.IFormatProvider is not used in the implementation of this method. [Note: There is no System.IFormatProvider that corresponds to a System.Enum object; therefore, <i>provider</i> is not utilized by this method, and any System.IFormatProvider can be passed as a parameter.]

12 13 Return Value

14 The System.String representation of the value of the current instance as specified by
15 *format*.

16 Description

17 If *format* is a null reference or an empty string, the return value is formatted using the
18 general format specifier ("G").

19
20 [Note: For details on the format specifiers used with an enumeration object, see
21 System.Enum.Format.

22
23 This method is implemented to support the System.IFormattable interface.

24
25]

26 Exceptions

Exception	Condition
System.FormatException	<i>format</i> does not contain a valid format specifier.

1

2

1 Enum.ToString(System.IFormatProvider)

2 Method

```
3 [ILAsm]  
4 .method public final hidebysig virtual string ToString(class  
5 System.IFormatProvider provider)  
  
6 [C#]  
7 public string ToString(IFormatProvider provider)
```

8 Summary

9 Converts the name of the value of the current instance to its equivalent System.String
10 representation.

11 Parameters

Parameter	Description
<i>provider</i>	An object that implements the System.IFormatProvider interface or a null reference. The System.IFormatProvider is not used in the implementation of this method. [Note: There is no System.IFormatProvider that corresponds to a System.Enum object; therefore, <i>provider</i> is not utilized by this method, and any System.IFormatProvider can be passed as a parameter.]

12

13 Return Value

14 The System.String representation of the name of the value of the current instance.

15 Description

16 This method is equivalent to the version of System.Enum.ToString that takes no
17 arguments.

18