

System.Runtime.InteropServices.CallingConvention Enum

```
[ILASM]
.class public sealed serializable CallingConvention extends
System.Enum

[C#]
public enum CallingConvention
```

Assembly Info:

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

Summary

Indicates the calling convention used by a method located in an unmanaged shared library.

Inherits From: System.Enum

Library: RuntimeInfrastructure

Description

The values of this enumeration are used to specify the calling conventions required to call unmanaged methods implemented in shared libraries.

[*Note:* Implementers should map the semantics of specified calling conventions onto the calling conventions of the host OS.]

[*Note:* For additional information on shared libraries and an example of the use of the **System.Runtime.InteropServices.CallingConvention** enumeration, see the **System.Runtime.InteropServices.DllImportAttribute** class overview.]

CallingConvention.Cdecl Field

```
[ILASM]  
.field public static literal valuetype  
System.Runtime.InteropServices.CallingConvention Cdecl = 2  
  
[C#]  
Cdecl = 2
```

Summary

Indicates that the **cdecl** calling convention is appropriate for a method call.

For example, on a Windows platform the **System.Runtime.InteropServices.CallingConvention.Cdecl** convention produces the following behavior:

Element	Behavior
Argument-passing order	Right to left.
Stack-maintenance responsibility	Calling function pops the arguments from the stack.

[Note: This is the default calling convention for functions compiled with 32-bit C and C++ language compilers.]

1 CallingConvention.FastCall Field

```
2 [ILASM]  
3 .field public static literal valuetype  
4 System.Runtime.InteropServices.CallingConvention FastCall =  
5 5  
  
6 [C#]  
7 FastCall = 5
```

8 Summary

9 Indicates that the **fastcall** calling convention is appropriate for a
10 method call.

11
12 [Note: On a Windows platform this convention indicates that
13 arguments to functions are to be passed in registers whenever
14 possible.]

15

1 CallingConvention.StdCall Field

```
2 [ILASM]
3 .field public static literal valuetype
4 System.Runtime.InteropServices.CallingConvention StdCall =
5 3
6
7 [C#]
8 StdCall = 3
```

8 Summary

9 Indicates that the **stdcall** calling convention is appropriate for a
10 method.

11
12 For example, on a Windows platform the
13 **System.Runtime.InteropServices.CallingConvention.StdCall**
14 convention produces the following behavior:

Element	Behavior
Argument-passing order	Right to left.
Stack-maintenance responsibility	Called function pops its own arguments from the stack.

15

16

1 CallingConvention.ThisCall Field

```
2 [ILASM]
3 .field public static literal valuetype
4 System.Runtime.InteropServices.CallingConvention ThisCall =
5 4
6
7 [C#]
8 ThisCall = 4
```

8 Summary

9 Indicates that the **thiscall** calling convention is appropriate for a
10 method. This convention is similar to the
11 **System.Runtime.InteropServices.CallingConvention.Cdecl**
12 calling convention, except that the last element that the caller pushes
13 the stack is the **this** pointer.

14
15 For example, on a Windows platform the
16 **System.Runtime.InteropServices.CallingConvention.ThisCall**
17 convention produces the following behavior:

Element	Behavior
Argument-passing order	Right to left.
Stack-maintenance responsibility	Calling function pops the arguments from the stack.
this pointer	Pushed last onto the stack.

18
19 [Note: The **thiscall** calling convention is the default calling convention
20 used by C++ member functions that are not called with a variable
21 number of arguments.]

22