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 00]
- *Version:* 2.0.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.]

CallingConvention.FastCall Field

[ILAsm]
```
.field public static literal valuetype
System.Runtime.InteropServices.CallingConvention FastCall = 5
```

[C#]
```
FastCall = 5
```

Summary

Indicates that the `fastcall` calling convention is appropriate for a method call.

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

CallingConvention.StdCall Field

[ILAsm]
```
.field public static literal valuetype
System.Runtime.InteropServices.CallingConvention StdCall = 3
```

[C#]
```
StdCall = 3
```

Summary

Indicates that the `stdcall` calling convention is appropriate for a method.

For example, on a Windows platform the `System.Runtime.InteropServices.CallingConvention.StdCall` 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.

CallingConvention.ThisCall Field

[ILAsm]
```
.field public static literal valuetype
System.Runtime.InteropServices.CallingConvention ThisCall = 4
```

[C#]
```
ThisCall = 4
```

Summary

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

For example, on a Windows platform the `System.Runtime.InteropServices.CallingConvention.ThisCall` 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.

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