

Calling a DLL Function

Article • 03/11/2022

Although calling unmanaged DLL functions is nearly identical to calling other managed code, there are differences that can make DLL functions seem confusing at first. This section introduces topics that describe some of the unusual calling-related issues.

Structures that are returned from platform invoke calls must be data types that have the same representation in managed and unmanaged code. Such types are called *blittable types* because they do not require conversion (see [Blittable and Non-Blittable Types](#)). To call a function that has a non-blittable structure as its return type, you can define a blittable helper type of the same size as the non-blittable type and convert the data after the function returns.

In This Section

[Passing Structures](#)

Identifies the issues of passing data structures with a predefined layout.

[Callback Functions](#)

Provides basic information about callback functions.

[How to: Implement Callback Functions](#)

Describes how to implement callback functions in managed code.


Related Sections

[Consuming Unmanaged DLL Functions](#)

Describes how to call unmanaged DLL functions using platform invoke.

[Marshalling Data with Platform Invoke](#)

Describes how to declare method parameters and pass arguments to functions exported by unmanaged libraries.

 Collaborate with us on
GitHub

The source for this content can be found on GitHub, where you can also create and review



.NET feedback

.NET is an open source project.
Select a link to provide feedback:

issues and pull requests. For more information, see [our contributor guide](#).



[Open a documentation issue](#)



[Provide product feedback](#)