**Choose between stub and shim types**

Typically, you would consider a Visual Studio project to be a component, because you develop and update those classes at the same time. You would consider using stubs and shims for calls that the project makes to other projects in your solution, or to other assemblies that the project references.

As a general guide, use stubs for calls within your Visual Studio solution, and shims for calls to other referenced assemblies. This is because within your own solution it is good practice to decouple the components by defining interfaces in the way that stubbing requires. But external assemblies such as *System.dll* typically are not provided with separate interface definitions, so you must use shims instead.

Other considerations are:

**Performance.** Shims run slower because they rewrite your code at run time. Stubs do not have this performance overhead and are as fast as virtual methods can go.

**Static methods, sealed types.** You can only use stubs to implement interfaces. Therefore, stub types cannot be used for static methods, non-virtual methods, sealed virtual methods, methods in sealed types, and so on.

**Internal types.** Both stubs and shims can be used with internal types that are made accessible by using the assembly attribute [InternalsVisibleToAttribute](https://docs.microsoft.com/en-us/dotnet/api/system.runtime.compilerservices.internalsvisibletoattribute).

**Private methods.** Shims can replace calls to private methods if all the types on the method signature are visible. Stubs can only replace visible methods.

**Interfaces and abstract methods.** Stubs provide implementations of interfaces and abstract methods that can be used in testing. Shims can't instrument interfaces and abstract methods, because they don't have method bodies.

In general, we recommend that you use stub types to isolate from dependencies within your codebase. You can do this by hiding the components behind interfaces. Shim types can be used to isolate from third-party components that do not provide a testable API.