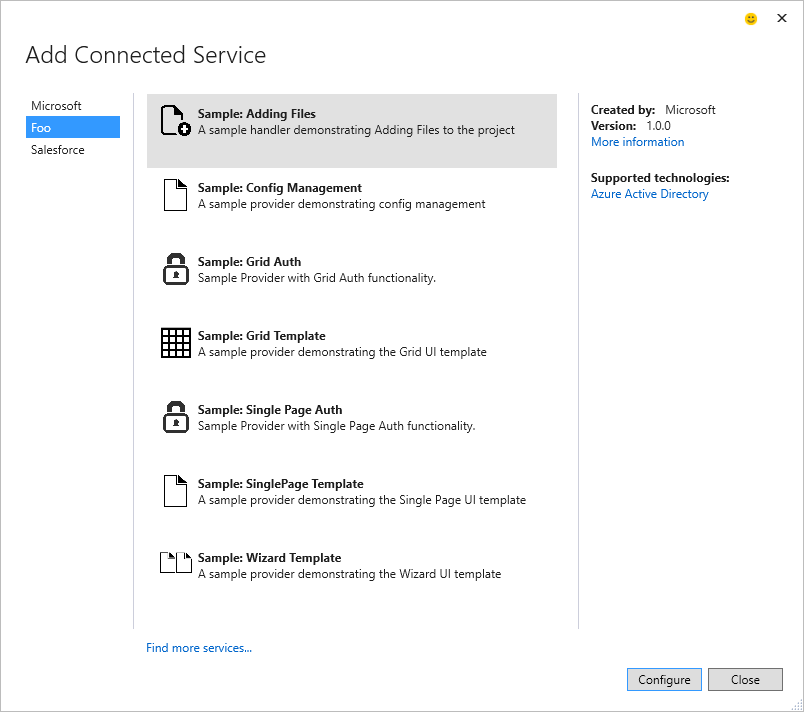
Troubleshooting Visual Studio Extensions and Connected Service Providers

# Conflicting Providers & Extensions

If you’ve completely uniquefied your providers, you can load any number at the same time. The following documents a number of gotchas to look for to assure your Visual Studio Extensions and Connected Services Providers are properly unique.



Visual Studio Extensions (VSIX) must be resolved as independent components. When using existing projects, and attempting to rename a few things, it's important to know which elements make up the unique name so you don't get a conflict when attempting to load what should be two extensions, but Visual Studio doesn't realize they are different.

Unique Location on Disk

As an example, the Microsoft Sample for the Grid Template extension is placed in the following directory:

%userprofile%\APPDATA\LOCAL\MICROSOFT\VISUALSTUDIO\**14.0EXP**\EXTENSIONS\**MICROSOFT\Microsoft Connected Service Sample Grid**\**1.0**

This could be tokenized as:

%userprofile%\APPDATA\LOCAL\MICROSOFT\VISUALSTUDIO\**14.0EXP**\EXTENSIONS\**[Manifest Author]\[Manifest Product Name]**\**[Manifest Version]**

If you've used a template that you've already installed in your primary or Experimental instance of VS, but have only changed some code, you may likely get a conflict.

Be sure to use the Manifest designer to rename the Author, Product Name and Version of your extension

Unique Binary Names

The Extension Manager will load your assemblies in a single app domain. If you've forgotten to rename the Assembly name from Project Properties, you may find strange errors. Visual Studio will report errors when MEF attempts to load the extensions, and if you look closely, the error may be pointing at the "other" dll that uses the same assembly name.

**To resolve this**, simply open the Project Properties and check update the assembly name.

**Troubleshooting**: By looking in %userprofile%\APPDATA\LOCAL\MICROSOFT\VISUALSTUDIO\**14.0EXP**\EXTENSIONS\, look in each extension directory and make sure the assembly names are unique across all the extensions.

Connected Service Provider/Handler Names

Be sure your project has unique names for the ProviderID. These of course should be the same across the matching Provider and Handers:

Provider

[Export(typeof(ConnectedServiceProvider))]

[ExportMetadata("ProviderId", "Microsoft.ConnectedServiceSamples.FooService.SinglePage")]

internal class FooProvider : ConnectedServiceProvider

Handler

[Export(typeof(ConnectedServiceHandler))]

[ExportMetadata("ProviderId", "Microsoft.ConnectedServiceSamples.FooService.SinglePage")]

[ExportMetadata("AppliesTo", "CSharp")]

internal class FooHander : ConnectedServiceHandler

Reference

General Troubleshooting VSIX Loading Errors

Review the following file. Look at the last few entries as you'll see extensions loaded, with a series of failures.

%userprofile%\AppData\Roaming\Microsoft\VisualStudio\14.0Exp\ActivityLog.xml

[VSIX Manifest Designer](https://msdn.microsoft.com/en-us/library/ee943167.aspx)

[VSIX Extension Schema 2.0 Reference](https://msdn.microsoft.com/en-us/library/hh696828.aspx)

[Anatomy of a VSIX Package](https://msdn.microsoft.com/en-us/library/dd997148.aspx)