Desktop Framework Bloat with Main Rx Package

Rx versions for which use of a Windows-specific TFM for 10.0.19041 or higher (e.g. `net8.0-windows10.0.19041`) causes self-contained deployments to contain a copy of the WPF and Windows Forms frameworks even when the application does not use these.

Note: this addresses only direct use of the relevant Rx version's main package. Scenarios where applications end up with transitive dependencies on older versions of Rx can continue to cause bloat with versions that don't appear here.

RequestedDesktopFramework	DeployedDesktopFramework	config.rxVersion.version	config.EmitDisableTransitiveFrameworkReferences
False	True	5.0.0	False
False	True	6.0.1	False

config.rxVersion.version

- □ 3.0.0
- □ 3.1.0
- □ 4.4.1
- □ 5.0.0
- □ 6.0.1
- □ 7.0.0-preview-legacyfacade.1.ga1159cd7f3
- □ 7.0.0-preview-legacyfacade-refnoui.3.g7492bd5...
- □ 7.0.0-preview-legacyfacade-refnoui-withfxref.1.g...
- □ 7.0.0-preview-nofacade-refnoui.5.gc59ebd3e22

Plug-Ins Get Wrong Rx Target

This shows scenarios in which plug-ins can end up getting an Rx assembly for the wrong target if they end up loading after another plug-in that uses the same version of Rx.

This only afflicts .NET Framework plug-in hosts, because .NET can use a separate assembly resolution context for each plug-in.

Note that Rx 3.1 fixed this problem, but the fix was removed in Rx 4 (possibly under the mistaken belief that the Great Unification would also prevent this problem) but it happened not to cause a visible problem at the time because Rx 4 happened not to offer a combination of targets that would enable this problem to occur in practice. But Rx 5's set of targets did enable it to happen, meaning the absence of a fix for this has meant that this problem continues to this day.

config.rxVersion.version

- □ 3.0.0
- □ 3.1.0
- □ 4.4.1
- □ 5.0.0
- □ 6.0.1
- □ 7.0.0-preview-legacyfacade.1.ga1159cd7f3
- □ 7.0.0-preview-legacyfacade-refnoui.3.g7492bd5...
- □ 7.0.0-preview-legacyfacade-refnoui-withfxref.1.g...
- □ 7.0.0-preview-nofacade-refnoui.5.gc59ebd3e22

config.rxVersion.version	config.plugIn1.tfm	config.plugIn2.tfm	plugIn1.rxTargetFramework	plugIn2.rxTargetFramework
3.0.0	net45	net46	.NETFramework,Version=v4.5	.NETFramework,Version=v4.5
3.0.0	net46	net45	.NETFramework,Version=v4.6	.NETFramework,Version=v4.6
5.0.0	net462	net472	.NETStandard,Version=v2.0	.NETStandard,Version=v2.0
5.0.0	net472	net462	.NETFramework,Version=v4.7.2	.NETFramework,Version=v4.7.2
6.0.1	net462	net472	.NETStandard,Version=v2.0	.NETStandard,Version=v2.0
6.0.1	net472	net462	.NETFramework,Version=v4.7.2	.NETFramework,Version=v4.7.2
7.0.0-preview-legacyfacade.1.ga1159cd7f3	net462	net472	.NETStandard,Version=v2.0	.NETStandard,Version=v2.0
7.0.0-preview-legacyfacade.1.ga1159cd7f3	net472	net462	.NETStandard,Version=v2.0	.NETStandard,Version=v2.0
7.0.0-preview-legacyfacade-refnoui.3.g7492bd514e	net462	net472	.NETStandard,Version=v2.0	.NETStandard,Version=v2.0
7.0.0-preview-legacyfacade-refnoui.3.g7492bd514e	net472	net462	.NETStandard,Version=v2.0	.NETStandard,Version=v2.0
7.0.0-preview-legacyfacade-refnoui-withfxref.1.g307e3b7f27	net462	net472	.NETStandard,Version=v2.0	.NETStandard,Version=v2.0
7.0.0-preview-legacyfacade-refnoui-withfxref.1.g307e3b7f27	net472	net462	.NETStandard,Version=v2.0	.NETStandard,Version=v2.0
7.0.0-preview-nofacade-refnoui.5.gc59ebd3e22	net462	net472	.NETStandard,Version=v2.0	.NETStandard,Version=v2.0
7.0.0-preview-nofacade-refnoui.5.gc59ebd3e22	net472	net462	.NETFramework,Version=v4.7.2	.NETFramework,Version=v4.7.2

Build failures due to UI-Framework-Specific Extension Methods

Since the 'bloat' problem involves an application acquiring an unwanted dependency on the `Microsoft.Desktop.App` framework (i.e. WPF and Windows Forms), one obvious response to this is to attempt to block that framework reference while minimizing other changes.

However, this can cause new problems. If the UI-framework-specific code is still present, the compiler can see it. And if the application tries to use an overload for which a UI-framework-specific overload exists, then even if it's not actually trying to use one of those overloads (and is instead using one of the overloads that's available in all targets), the compiler reports an error because it can see these overloads but can't load the UI framework types that these overloads refer to, and so it can't be certain that the method invocation isn't ambiguous.

As the table shows below, if we try to prevent bloat with Rx 5 or 6 by just blocking the framework dependency (by setting the EmitDisableTransitiveFrameworkReferences project property to True) this problem occurs.

There are possible packaging designs for future versions of Rx that would have the same problem even without EmitDisableTransitiveFrameworkReferences , which is why we test for this.

config.rxVersion.version	config.EmitDisableTransitiveFrameworkReferences
5.0.0	True
6.0.1	True

config.rxVersion.version

- □ 3.0.0
- □ 3.1.0
- □ 4.4.1
- □ 5.0.0
- □ 6.0.1
- □ 7.0.0-preview-legacyfacade.1.ga1159cd7f3
- □ 7.0.0-preview-legacyfacade-refnoui.3.g7492bd5…
- □ 7.0.0-preview-legacyfacade-refnoui-withfxref.1.g…
- □ 7.0.0-preview-nofacade-refnoui.5.gc59ebd3e22