Set up your desktop application for MSIX packaging in Visual Studio

Article • 08/17/2022 • 2 minutes to read

You can use the **Windows Application Packaging Project** project in Visual Studio to generate a package for your desktop app. Then you can distribute your package to the Microsoft Store, on the Web, in your enterprise or any other distribution mechanism you're using.

Required Visual Studio version and workload

The **Windows Application Packaging Project** project is available in the following versions of Visual Studio:

- Visual Studio 2019
- Visual Studio 2017 15.5 and later

To see the Windows Application Packaging Project template in the 'Add New Project' menu, you need to make sure you have **at least one** of the following the Visual Studio workloads installed:

- The 'Universal Windows Platform development' workload
- The Optional Component 'MSIX Packaging Tools' in the NET Core workload.
- The Optional Component 'MSIX Packaging Tools' in the .NET desktop development workload.

For the best experience we recommend that you use the latest Visual Studio release.

> **ⓘ Important**
>
> The **Windows Application Packaging Project** project in Visual Studio is supported on Windows 10, version 1607, and later. It can only be used in projects that target Windows 10 Anniversary Update (10.0; Build 14393) or a later release.

Here are a few other things you can do from the Visual Studio Application Packaging Project:

✔ Automatically generate visual assets.

✔ Make changes to your manifest using a visual designer.

✔ Generate your package or bundle using a wizard.

✔ (If publishing to the Microsoft Store) Easily assign an identity to your application from a name that you've already reserved in Partner Center .

Prepare your application

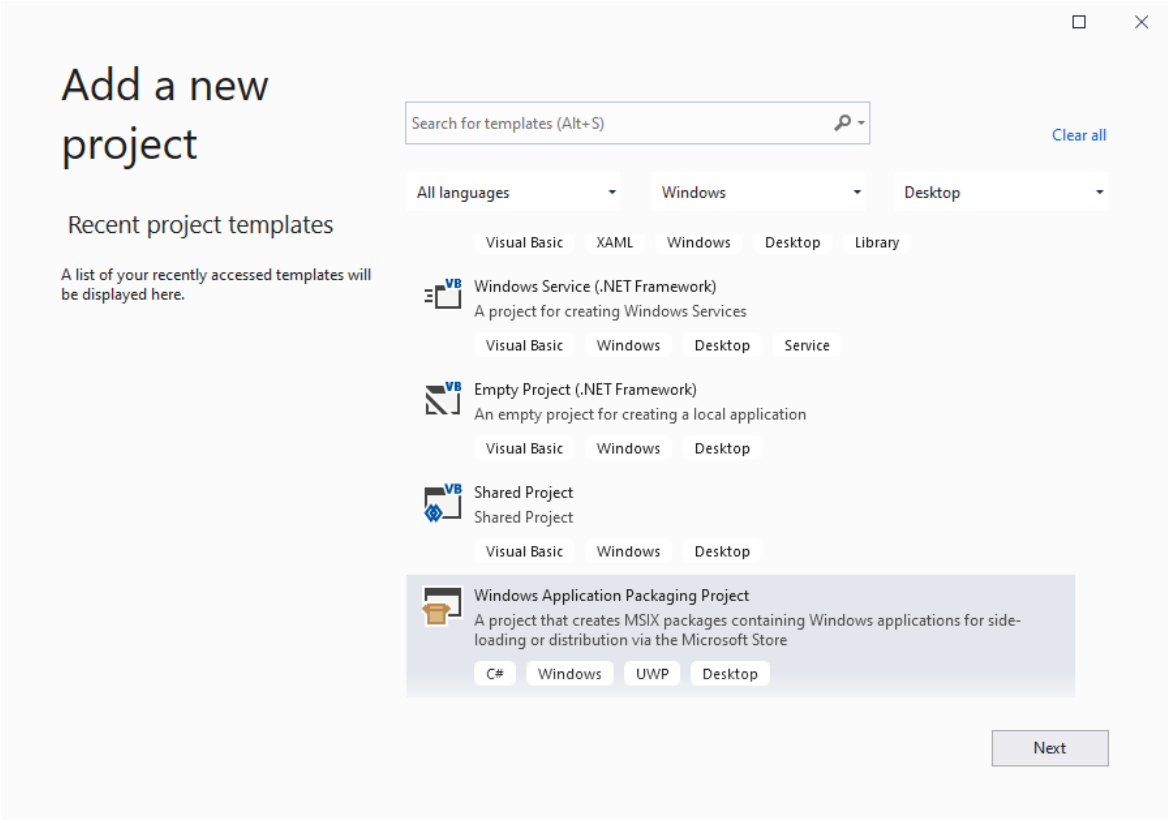
Review this guide before you begin creating a package for your application: Prepare to package a desktop application.

Setup the Windows Application Packaging Project in your solution

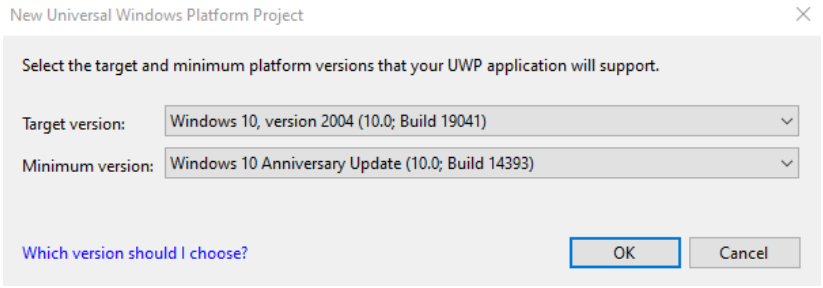
The screenshots below are from Visual Studio 2019 16.10.

1. In Visual Studio, open the solution that contains your desktop application project.
2. Add a **Windows Application Packaging Project** project to your solution.

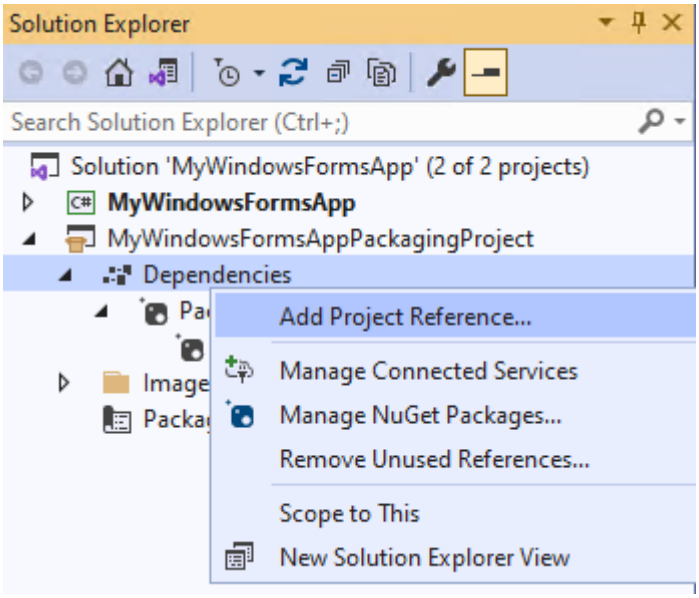
   You won't have to add any code to it. It's just there to generate a package for you. We'll refer to this project as the "packaging project".



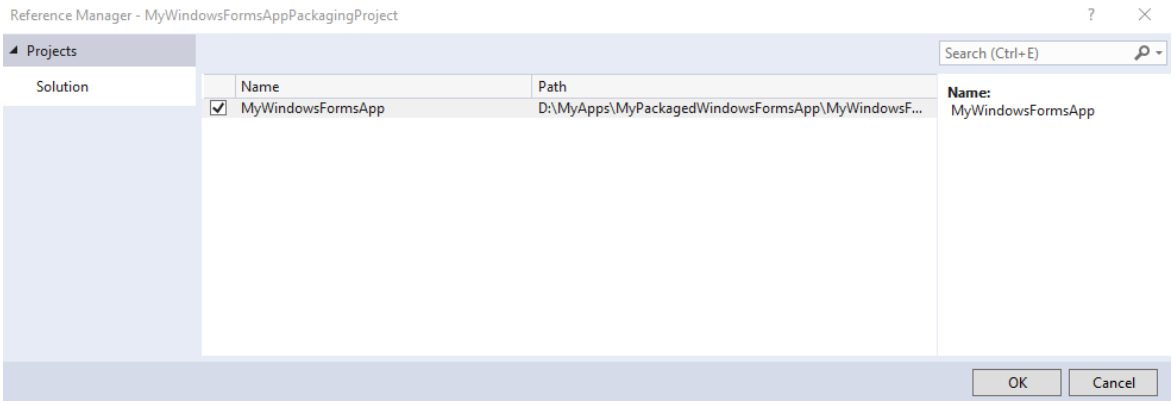
3. Set the **Target Version** of this project to any version that you want, but make sure to set the **Minimum Version** to not lower than **Windows 10 Anniversary Update**.



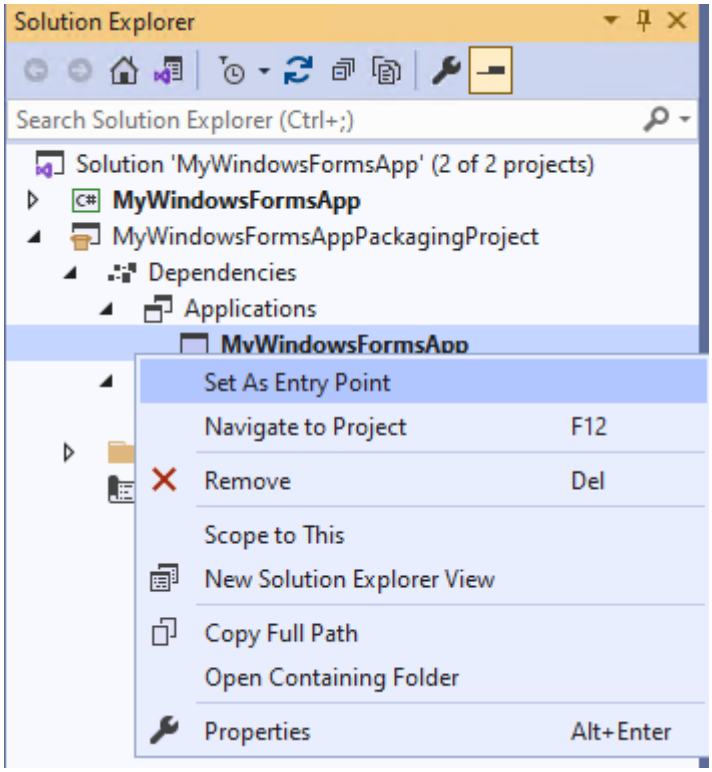
4. In Solution Explorer, right-click the **Dependencies** folder under the packaging project and choose **Add Project Reference...**.



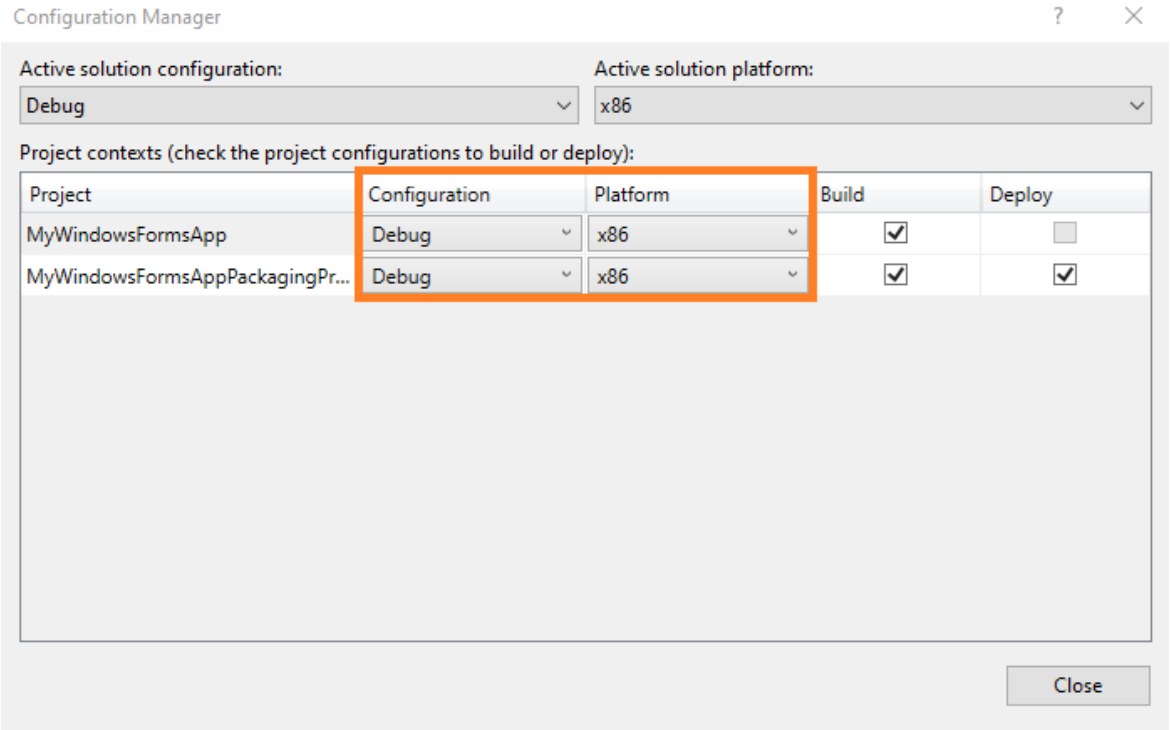
5. Choose your desktop application project, and then choose the **OK** button.



You can include multiple desktop applications in your package, but only one of them can start when users choose your app tile. In the **Applications** node, right-click the application that you want users to start when they choose the app's tile, and then choose **Set as Entry Point**.



6. Build the packaging project to ensure that no errors appear. If you receive errors, open **Configuration Manager** and ensure that your projects target the same platform.



7. Use the Create App Packages wizard to generate an MSIX package/bundle or an .msixupload/.appxupload file (for Store publishing to the Store).

Next steps

Package your desktop app in Visual Studio

See Package a Desktop or UWP app in Visual Studio

Run, debug or test your desktop application

See Run, debug, and test a packaged application

Additional resources

Enhance your desktop application by adding UWP APIs

See Enhance your desktop application for Windows 10

Extend your desktop application by adding UWP projects and Windows Runtime Components

See Extend your desktop application with modern UWP components.

Distribute your app

See Distribute a packaged desktop application