How can I change a .NET standard library to a .NET framework library?

Ask Question

8

1

I'm writing a class library for a simple parser in C#. When I first created it, I used .NET standard 2.0, but now I need to migrate it to .NET 4.6 both to conform to the other projects in my solution and in order to use NUnit.

I tried to follow the instructions in the Microsoft documentation, but when I try to select another framework in the properties, I can only find other .NET standard versions.

How can I migrate it? Will I need to manually edit the `.csproj` file?

c# .net .net-standard .net-framework-version .net-4.6

asked Jul 5 '18 at 14:28

Edward Minnix
1,246 7 18

Which version of Visual Studio are you using? – j03p Jul 5 '18 at 14:32

Visual Studio Professional 2017 – Edward Minnix Jul 5 '18 at 14:34

It is interesting that you said "now I need to migrate it to .NET 4.6 both to conform to the other projects in my solution and in order to use NUnit". Why do you need to migrate? Typically NUnit can work well with .NET

Home

PUBLIC

Stack Overflow

Tags

Standard (though the assembly hosting NUnit test cases might need to target net46). – Lex Li Jul 5 '18 at 14:40

The NUnit docs say that the assembly that the tests are in either have to be .NET framework or .NET core. 4.6 is because there's 6 projects in the solution, and mine is only one of them (all the others are 4.6) – Edward Minnix Jul 5 '18 at 14:44

2 It is the point of .netstandard to not have to do this. Why anybody would target a framework version that was quite buggy and quickly had to be updated to 4.6.1 is a fair mystery. Fix that instead, take it to 4.7.1 while you're at it. – Hans Passant Jul 5 '18 at 14:45

4 Answers

12

Open up the project file (.csproj) and change the TargetFramework to net462

```
<PropertyGroup>
  <TargetFramework>net462</TargetFramework>
</PropertyGroup>
```

answered Jul 5 '18 at 14:30

Jammer
5,653 5 47 81

I used `net46` instead of `net462`, but it fixed NUnit, thanks! – Edward Minnix Jul 5 '18 at 14:37

No problem. Mark as answer if it fixed for you. Thanks. – Jammer Jul 5 '18 at 14:37

My personal experience in Visual Studio 2017 is that **recreating**

2

project and adding existent sources is the simplest, safest and most effective way - because .Net Framework based csproj file has extra xml elements (comparing with Standard based), it seems changing "TargetFramework" is not enough. Below is portion of diffs appeared by default:

```
-<Project Sdk="Microsoft.NET.Sdk">
-
+<?xml version="1.0" encoding="utf-8"?>
+<Project ToolsVersion="15.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
+  <Import Project="$(MSBuildExtensionsPath)\$(MSBuildToolsVersion)\Microsoft.Common.props" Co
   <PropertyGroup>
-    <TargetFramework>netstandard2.0</TargetFramework>
-    <RootNamespace>PipeWindowsService.ServiceUtilities</RootNamespace>
+    <Configuration Condition=" '$(Configuration)' == '' ">Debug</Configuration>
+    <Platform Condition=" '$(Platform)' == '' ">AnyCPU</Platform>
+    <ProjectGuid>{0CDC403B-AB6A-4F28-B874-F8BDAE3EB1C7}</ProjectGuid>
+    <OutputType>Library</OutputType>
+    <AppDesignerFolder>Properties</AppDesignerFolder>
+    <RootNamespace>ServiceUtilities</RootNamespace>
+    <AssemblyName>ServiceUtilities</AssemblyName>
+    <TargetFrameworkVersion>v4.7.1</TargetFrameworkVersion>
+    <FileAlignment>512</FileAlignment>
   </PropertyGroup>
-
+  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Debug|AnyCPU' ">
+    <DebugSymbols>true</DebugSymbols>
+    <DebugType>full</DebugType>
+    <Optimize>false</Optimize>
+    <OutputPath>bin\Debug\</OutputPath>
+    <DefineConstants>DEBUG;TRACE</DefineConstants>
+    <ErrorReport>prompt</ErrorReport>
+    <WarningLevel>4</WarningLevel>
+  </PropertyGroup>
+  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Release|AnyCPU' ">
+    <DebugType>pdbonly</DebugType>
+    <Optimize>true</Optimize>
+    <OutputPath>bin\Release\</OutputPath>
+    <DefineConstants>TRACE</DefineConstants>
+    <ErrorReport>prompt</ErrorReport>
```

edited Aug 13 '18 at 0:15

answered Aug 13 '18 at 0:07

Kaponir

831 6 20

This guy is right. There are much more of changes than in accepted answer. – Andrew Jan 4 at 1:56

2

If you are publishing your class library as a Nuget package then there is a better way to set this up. Check out this article:

https://weblog.west-wind.com/posts/2017/Jun/22/MultiTargeting-and-Porting-a-NET-Library-to-NET-Core-20

Basically you can setup your class library for multi targeting, allowing it to be imported into .net core projects as well as different versions of .net frameworks.

answered Jul 5 '18 at 14:54

Marko
7,369 5 31 43

0

There are a few steps that I did and worked for me:

1. `git push` your code, so you have a back up :)
2. Unload the project in VS (right click on the project and unload)
3. Edit the project in VS (right click and edit)
4. Replace the TargetFramework OR/AND TargetFrameworkVersion with
   `<TargetFramework>netcoreapp2.0</TargetFramework>`
5. Change the project line, that's usually the first line (after xml root) to: `<Project Sdk="Microsoft.NET.Sdk" >`
6. Remove the import that's usually the second line (after the xml root)
7. Keep your PropertyGroups that describe the build options, if you want (I want mine as are custom)
8. Remove the references and the file references, they are not needed.
9. Close the file and reload (right click reload).

10. Delete the assemblyinfo file (from properties folder) as it is not needed, the assembly version comes now from the proj
11. Right click on the project and go to properties to see that you don't have any error in proj file. Ensure that you don't have typos or tags that are not close.
12. Build. If you have dependencies that you are missing then right click and on the project and add them. - I suppose that you don't want to edit them in the proj. Or you can do a dotnet restore or dotnetbulid to add them, as you would like.

Hope this works for you. They seem a lot of steps but they are not that complicated, and this is one time effort.

edited Feb 27 at 15:30

Community ♦
1 1

answered Oct 18 '18 at 3:38

Dan
1 1