This project references NuGet package(s) that are missing on this computer

Ask Question

266

I have an ASP.NET MVC5 application that worked yesterday and now I am getting this error when I try to build:

> This project references NuGet package(s) that are missing on this computer.

56

I have the two options checked that allow nuget to automatically download and install missing packages checked / turned ON. I have also tried deleting all of the files in the packages folder and then have nuget re-download them. Also when I open nuget and look for updates it says there are none that need to be installed. I can't figure what else to do to move beyond this amazingly annoying issue.

visual-studio-2010 visual-studio msbuild nuget

edited Jan 23 '17 at 8:57

Barry Michael Doyle
2,732 8 44 75

asked Apr 7 '14 at 10:14

Austin Harris
1,852 5 18 34

Home

PUBLIC

Stack Overflow

Tags

Users

three items in that folder and does nothing to resolve the problem. I've tried re-building and still get the same error above. – Austin Harris Apr 7 '14 at 10:24

Does your solution includes a .nuget folder and have you updated NuGet to latest version? See here: stackoverflow.com/questions/18833649/… – David Brabant Apr 7 '14 at 10:27

Yes, tried that and it did not solve my build error message problem. – Austin Harris Apr 7 '14 at 18:01

Another reason for this error is an `The operation has timed out.` error. during the build. You need to check your build log, or the **Diagnostics** tab in the Visual Studio Online Build Failed information screen. – Joshua Drake Oct 22 '15 at 18:15

None of the solutions work for me. I'm downloading from a repo and the packages restore in the correct file structure for the first project, second project can't find them. Checking the .csproj shows that the correct relative path is being used so I'm at a loss for trying to solve this. github.com/DanJ210/ProgrammersTest – Daniel Jackson Apr 8 at 16:50

15 Answers

406

In my case, I had to remove the following from the .csproj file:

```
<Import Project="$(SolutionDir)\.nuget\NuGet.targets"
Condition="Exists('$(SolutionDir)\.nuget\NuGet.targets')" /
<Target Name="EnsureNuGetPackageBuildImports" BeforeTarget
  <PropertyGroup>
    <ErrorText>This project references NuGet package(s) th
computer. Enable NuGet Package Restore to download them.
http://go.microsoft.com/fwlink/?LinkID=322105. The missing
  </PropertyGroup>
  <Error Condition="!Exists('$(SolutionDir)\.nuget\NuGet.t
```

In fact, in this snippet you can see where the error message is coming from.

I was converting from **MSBuild-Integrated Package Restore** to **Automatic Package Restore** (http://docs.nuget.org/docs/workflows/migrating-to-automatic-package-restore)

edited Aug 26 '14 at 11:27

Mikaël Mayer
6,898 3 49 84

answered May 25 '14 at 4:59

Loren Paulsen
6,550 1 23 34

7 Thanks, finally an answer that contains all the lines that need to be deleted. Took me many hours. If you have a solution that contains multiple projects, then the error message row contains the name of the project where you should apply this. – Albert Jul 13 '14 at 14:57

Thank you very much. For Empty Website Projects this is a must, to get the project up-and-running again after "Nuget Packages are missing". – Jorrit Reedijk Jul 31 '14 at 19:52

6 This worked for me, but I only needed to remove the <Target></Target> element. VS[2013] seemed to restore it if I removed the <Import> element as well. – Robert Taylor Dec 18 '14 at 10:52

7 if this can be removed, why is it there at the first place? – OK999 Aug 17 '16 at 19:43

1 OK9999, at one point, you must have enabled it from an earlier version of Visual Studio, by right clicking on the solution and choosing "Enable NuGet package restore" which did it the old way. We don't need that anymore – Loren Paulsen Aug 18 '16 at 19:19

67

One solution would be to remove from the .csproj file the following:

```
<Import Project="$(SolutionDir)\.nuget\NuGet.targets"
Condition="Exists('$(SolutionDir)\.nuget\NuGet.targets')" ,
<Target Name="EnsureNuGetPackageBuildImports" BeforeTarget
  <PropertyGroup>
    <ErrorText>This project references NuGet package(s) tha
computer. Enable NuGet Package Restore to download them.  
http://go.microsoft.com/fwlink/?LinkID=322105. The missing
  </PropertyGroup>
  <Error Condition="!Exists('$(SolutionDir)\.nuget\NuGet.ta
Text="$([System.String]::Format('$(ErrorText)', '$(Solutio
/>
</Target>
```

How?

1. Right click on project. Unload Project.
2. Right click on project. Edit csproj.
3. Remove the part from the file. Save.
4. Right click on project. Reload Project.

answered Mar 28 '16 at 14:58

Ivan Santiago
1,028 14 13

This worked for me. Thanks! – SyntaxError Aug 3 '16 at 14:45

Thank you for this... This worked for me. I did not have to remove the <Import> but just the <Target Name="EnsureNugetPackageBuildImports" ...> – Jerrod Horton Aug 22 '16 at 15:26

3 @IvanSantiago **It was already ANSWERED above with the same solution!..** Vote DOWN!.. – Willys May 16 '18 at 8:50

1 @ClintEastwood My answer explained **HOW** to do it. That's the difference. If a user is looking for a **HOW TO** my answer

1 @IvanSantiago You could have: added it as a comment, or, edited the original answer with the How To. – Colin Dec 28 '18 at 23:50

41

In my case it happened after I moved my solution folder from one location to another, re-organized it a bit and in the process its relative folder structure changed.

So I had to edit all entries similar to the following one in my `.csproj` file from

```
  <Import
Project="..\packages\Microsoft.Bcl.Build.1.0.14\tools\Micr
Condition="Exists('..\packages\Microsoft.Bcl.Build.1.0.14\
 />
```

to

```
  <Import
Project="packages\Microsoft.Bcl.Build.1.0.14\tools\Microso
Condition="Exists('packages\Microsoft.Bcl.Build.1.0.14\too
 />
```

(Note the change from `..\packages\` to `packages\`. It might be a different relative structure in your case, but you get the idea.)

edited May 15 '17 at 5:00

answered Mar 9 '15 at 20:20

3 Similar issue....I had moved the .csproj file up a level in the directory structure and had to change from "..\..\packages\..." to "..\packages\...". – tmgirvin Jun 3 '15 at 23:20

2 I had a similar issue, but really weird. I was using it in a subsolution module, so it was fine in that solution, but when I referenced that solution from another solution, the packages were in a different place. I changed ..\packages to $(SolutionDir)packages throughout the .csproj and that fixed it. – JoeNCA Jul 15 '15 at 21:15

2 If you don't want to muck around with the .csproj file manually I've found that taking a note of all the nuget packages you have installed for the project, deleting them and reinstalling them resolved this issue for me. I was trying to remove a project from a solution to put into its own git repository when I ran into this issue. – WiteCastle Aug 26 '15 at 2:36

does this mean your .csproj is at the same level as your .sln file? – Simon_Weaver May 13 '17 at 2:35

@Simon_Weaver the position of your `.csproj` relative to your `.sln` does not matter in this context. What matters is whether anything that is referenced in your `.csproj` had moved somewhere else. If so, then you need to fix it. If you moved your '.csproj' with everything that it references intact, but kept your `.sln` where it was, then you'd have to fix the `.sln` file to the new location of `.csproj` -es, but there would be no need to edit `.csproj` files. – Nikita G. May 15 '17 at 4:58

21

I easily solve this problem by right clicking on my solution and then clicking on the **Enable NuGet Package Restore** option

(P.S: Ensure that you have the Nuget Install From Tools--> Extensions and Update--> Nuget Package Manager for

edited Mar 18 '15 at 8:32

answered Mar 18 '15 at 8:27

M.A
238 2 8

5 This is the old way of restoring nuget packages and should be avoided. – The Muffin Man Aug 25 '15 at 17:49

2 @TheMuffinMan: Can you please clarify what the new way is and why this way should be avoided (considering that VS 2013 error output tells you to do exactly that)? – CantrianBear Aug 19 '16 at 17:15

2 @CantrianBear Navigate to this page docs.nuget.org/consume/package-restore and find the section called `MSBuild-Integrated Package Restore` . That is the old way and it lists some reasons why you should use the new way. – The Muffin Man Aug 19 '16 at 17:58

See David Ebbo's blog on this blog.davidebbo.com/2014/01/… Now... "NuGet now always restores packages before building in VS." – timB33 Mar 4 at 9:54

17

In my case it had to do with the Microsoft.Build.Bcl version. My nuget package version was 1.0.21, but my project files were still pointing to version 1.0.14

So I changed my .csproj files from:

```
  <Import
Project="..\..\packages\Microsoft.Bcl.Build.1.0.14\tools\M
Condition="Exists('..\..\packages\Microsoft.Bcl.Build.1.0.
 />
   <Target Name="EnsureBclBuildImported" BeforeTargets="Be
Condition="'$(BclBuildImported)' == ''">
```

```
Enable NuGet Package Restore to download them.  For more i
http://go.microsoft.com/fwlink/?LinkID=317567." HelpKeywor
    <Error
Condition="Exists('..\..\packages\Microsoft.Bcl.Build.1.0.
 Text="The build restored NuGet packages. Build the projec
packages in the build. For more information, see http://go
LinkID=317568." HelpKeyword="BCLBUILD2002" />
  </Target>
```

to:

```
 <Import
Project="..\..\packages\Microsoft.Bcl.Build.1.0.21\build\M
Condition="Exists('..\..\packages\Microsoft.Bcl.Build.1.0.
 />
  <Target Name="EnsureBclBuildImported" BeforeTargets="Bef
Condition="'$(BclBuildImported)' == ''">
    <Error
Condition="!Exists('..\..\packages\Microsoft.Bcl.Build.1.0
 Text="This project references NuGet package(s) that are m
Enable NuGet Package Restore to download them.  For more i
http://go.microsoft.com/fwlink/?LinkID=317567." HelpKeywor
    <Error
Condition="Exists('..\..\packages\Microsoft.Bcl.Build.1.0.
 Text="The build restored NuGet packages. Build the projec
packages in the build. For more information, see http://go
LinkID=317568." HelpKeyword="BCLBUILD2002" />
```

And the build was working again.

answered Jan 5 '15 at 16:48

henkie14

909 9 7

If you are using TFS

9

Remove the `NuGet.exe` and `NuGet.targets` files from the solution's `.nuget` folder. Make sure the files themselves are also removed from the solution workspace. Retain the `NuGet.Config` file to continue to bypass adding packages to source control.

Edit each project file (e.g., .csproj, .vbproj) in the solution and remove any references to the `NuGet.targets` file. Open the project file(s) in the editor of your choice and remove the following settings:

```
<RestorePackages>true</RestorePackages>
...
<Import Project="$(SolutionDir)\.nuget\nuget.targets" />
...
<Target Name="EnsureNuGetPackageBuildImports" BeforeTarget
    <PropertyGroup>
        <ErrorText>This project references NuGet package(s
computer. Enable NuGet Package Restore to download them.
http://go.microsoft.com/fwlink/?LinkID=322105. The missing
    </PropertyGroup>
    <Error Condition="!Exists('$(SolutionDir)\.nuget\NuGet
Text="$([System.String]::Format('$(ErrorText)', '$(Solutio
/>
</Target>
```

If you are not using TFS

Remove the `.nuget` folder from your solution. Make sure the folder itself is also removed from the solution workspace.

Edit each project file (e.g., .csproj, .vbproj) in the solution and remove any references to the `NuGet.targets` file. Open the project file(s) in the editor of your choice and remove the following settings:

```
<RestorePackages>true</RestorePackages>
```

```
<Target Name="EnsureNuGetPackageBuildImports" BeforeTarget
    <PropertyGroup>
        <ErrorText>This project references NuGet package(s
computer. Enable NuGet Package Restore to download them.
http://go.microsoft.com/fwlink/?LinkID=322105. The missing
    </PropertyGroup>
    <Error Condition="!Exists('$(SolutionDir)\.nuget\NuGet
Text="$([System.String]::Format('$(ErrorText)', '$(Solutio
/>
</Target>
```

Reference: Migrating MSBuild-Integrated solutions to use Automatic Package Restore

answered Jul 6 '16 at 2:36

RAM
8,157 4 40 61

6

Is it possible that the packages have been restored to the wrong folder? Check that the paths in the csproj files are correct.

If they are different it could be caused by the packages now being restored to a different location. This could be caused by a NuGet.Config file being checked in specifying a node like this:

```
<add key="repositoryPath" value="..\..\Packages" />
```

The packages are being restored, by the projects are still looking at the old location.

answered Apr 7 '14 at 11:39

psych
3,327 2 23 62

of the files but I don't see where there is a hard coded path anywhere. I looked in the proj file and all the packages files seem to be relative like this: <Reference Include="Antlr3.Runtime, Version=3.5.0.2, Culture=neutral, PublicKeyToken=eb42632606e9261f, processorArchitecture=MSIL"> <SpecificVersion>False</SpecificVersion> <HintPath>..\packages\Antlr.3.5.0.2\lib\Antlr3.Runtime.dll</HintPath> </Reference> – Austin Harris Apr 7 '14 at 17:57

adding this to web.config did not help: <add key="repositoryPath" value="..\..\Packages" /> – Austin Harris Apr 7 '14 at 22:55

You don't want to add that to the web.config. I was referring to the NuGet.config file and you want to check the relative path. Where are your packages relative to the csproj files? Are they in ..\packages? It sounds like the packages are being restored correctly, but your projects are looking in the wrong place. – psych Apr 8 '14 at 7:42

6

I had the same issue. In my case installing the ***Microsoft.Bcl.Build*** package fixed the problem.

answered Oct 2 '14 at 12:41

mheyman
2,903 27 28

This worked for me as well - but I have no idea whether the right thing to do was install that package (which has the same effect as henkie14's version changing answer below or just delete all those Targets - are they actually doing anything useful? – Gaz Feb 5 '15 at 10:15

I spent an hour on this issue before I read your answer – user2684198 Dec 28 '15 at 17:06

In `1.0.21` version no files in package, installation of `1.0.14`

1

These are the steps I used to fix the issue:

To add nuget packages to your solution:

1. Right click on the project (not solution) you want to reference nuget packages.
2. Choose: Manage nuget packages
3. On the popup window, on the left you have three choices. If you choose *Online* > *Microsoft & .NET*, you will be able to install Microsoft ASP.NET Web API 2.2 package grouper (or whatever package you need - mine was this).
4. Now right click on your solution (not project) and choose *Enable nuget package restore*. This will cause the packages to be automagically downloaded at compilation.

edited Feb 15 '15 at 21:47

gvlasov
6,961 16 50 86

answered Feb 15 '15 at 21:24

user4569838
11 1

All I had to do was enable nugget package restore for the solution. Apparently everything else was already set up correctly. – schmiddy98 Feb 23 '15 at 16:28

1

For me it worked as I just copied a .nuget folder from a working solution to the existing one, and referenced it's content!

meJustAndrew
3,258 4 26 53

0

I had the same issue when i reference the Class library into my MVC web application,

the issue was the nuget package version number mismatch between two projects.

ex: my class library had log4net of 1.2.3 but my webapp had 1.2.6

fix: just make sure both the project have the same version number referenced.

answered Nov 21 '14 at 9:26

Srini
328 6 22

0

Editing .sln and .csproj is not always that easy or desirable. Once you get the error list you can see what projects have missing packages (also, the References node usually indicate that there are missing assemblies unless packages are source code, resources, images, or just text-based ones).

Removing and then adding the packages is not a good idea unless you use the latest version of the package. Otherwise be prepared for surprises, not always pleasant ones.

If, say, the package is EntityFramework then from NuGet gallery you get the latest version which at the time of writing this comment it is **6.1.3**.

So, maybe the safest way to handle the situation is to restore the missing packages one by one. Yes, a bit painful exercise but chasing subtle bugs due to different package version maybe much more unpleasant.

Having this said, and let again EntityFramework be the missing package, you can issue the following command in the Package-Manager Console:

```
PM> Install-Package EntityFramework -Version 6.0.1
```

This will install the correct version, that is **6.0.1**, that is the one that is specified in packages.config:

```
<?xml version="1.0" encoding="utf-8"?>
<packages>
  <package id="EntityFramework" version="6.0.1" target
</packages>
```

edited Jul 2 '15 at 15:04

answered Jul 2 '15 at 14:58

Alexander Christov
2,810 1 24 42

0

Removed below lines in .csproj file

```
<Import Project="$(SolutionDir)\.nuget\NuGet.targets"
Condition="Exists('$(SolutionDir)\.nuget\NuGet.targets')"
<Target Name="EnsureNuGetPackageBuildImports" BeforeTarget
<PropertyGroup>
 <ErrorText>This project references NuGet package(s) that
```

```
<Error Condition="!Exists('$(SolutionDir)\.nuget\NuGet.tar
Text="$([System.String]::Format('$(ErrorText)',
'$(SolutionDir)\.nuget\NuGet.targets'))" />
</Target>
```

edited Mar 19 at 16:12

answered Mar 19 at 16:04

Amila Thennakoon
70 1 1 9

0

One solution would be to remove from the .csproj file the following:

```
<Import Project="$(SolutionDir)\.nuget\NuGet.targets"
Condition="Exists('$(SolutionDir)\.nuget\NuGet.targets')"
```

This project references NuGet package(s) that are missing on this computer. Enable NuGet Package Restore to download them. For more information, see http://go.microsoft.com/fwlink/?LinkID=322105. The missing file is {0}.

answered Apr 23 at 13:21

Umar Khaliq
1 2

I created a folder named '.nuget' in solution root folder
Then added file 'NuGet.Config' in this folder with following

-1

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
<solution>
 <add key="disableSourceControlIntegration" value="true" /
</solution>
</configuration>
```

Then created file '.nuGet.targets' as below
$(MSBuildProjectDirectory)..\

```
    <!-- Enable the restore command to run before builds -
    <RestorePackages Condition="  '$(RestorePackages)' ==

    <!-- Property that enables building a package from a pr
    <BuildPackage Condition=" '$(BuildPackage)' == '' ">fa

    <!-- Determines if package restore consent is required
    <RequireRestoreConsent Condition=" '$(RequireRestoreCo
">true</RequireRestoreConsent>

    <!-- Download NuGet.exe if it does not already exist -
    <DownloadNuGetExe Condition=" '$(DownloadNuGetExe)' ==
</PropertyGroup>

<ItemGroup Condition=" '$(PackageSources)' == '' ">
    <!-- Package sources used to restore packages. By defa
sources under %APPDATA%\NuGet\NuGet.Config -->
    <!--
        <PackageSource Include="https://nuget.org/api/v2/"
        <PackageSource Include="https://my-nuget-source/nu
    -->
</ItemGroup>

<PropertyGroup Condition=" '$(OS)' == 'Windows_NT'">
    <!-- Windows specific commands -->
    <NuGetToolsPath>$([System.IO.Path]::Combine($(SolutionD
</NuGetToolsPath>
    <PackagesConfig>$([System.IO.Path]::Combine($(ProjectDi
</PackagesConfig>
    <PackagesDir>$([System.IO.Path]::Combine($(SolutionDir
</PropertyGroup>

<PropertyGroup Condition=" '$(OS)' != 'Windows_NT'">
```

```
    <PackagesDir>$(SolutionDir)packages</PackagesDir>
</PropertyGroup>

<PropertyGroup>
    <!-- NuGet command -->
    <NuGetExePath Condition=" '$(NuGetExePath)' == ''
">$(NuGetToolsPath)\nuget.exe</NuGetExePath>
    <PackageSources Condition=" $(PackageSources) == '' ">(
</PackageSources>

    <NuGetCommand Condition=" '$(OS)' == 'Windows_NT'">"$(I
    <NuGetCommand Condition=" '$(OS)' != 'Windows_NT' ">mor
$(NuGetExePath)</NuGetCommand>

    <PackageOutputDir Condition="$(PackageOutputDir) == ''
</PackageOutputDir>

    <RequireConsentSwitch Condition=" $(RequireRestoreConse
RequireConsent</RequireConsentSwitch>
    <!-- Commands -->
    <RestoreCommand>$(NuGetCommand) install "$(PackagesCon
"$(PackageSources)"  $(RequireConsentSwitch) -o "$(Package
    <BuildCommand>$(NuGetCommand) pack "$(ProjectPath)" -p
Configuration=$(Configuration) -o "$(PackageOutputDir)" -sy

    <!-- Make the build depend on restore packages -->
    <BuildDependsOn Condition="$(RestorePackages) == 'true
        RestorePackages;
        $(BuildDependsOn);
    </BuildDependsOn>

    <!-- Make the build depend on restore packages -->
    <BuildDependsOn Condition="$(BuildPackage) == 'true'">
        $(BuildDependsOn);
        BuildPackage;
    </BuildDependsOn>
</PropertyGroup>

<Target Name="CheckPrerequisites">
    <!-- Raise an error if we're unable to locate nuget.exe
    <Error Condition="'$(DownloadNuGetExe)' != 'true' AND
Text="Unable to locate '$(NuGetExePath)'" />
    <SetEnvironmentVariable EnvKey="VisualStudioVersion"
EnvValue="$(VisualStudioVersion)" Condition=" '$(VisualStud
== 'Windows_NT' " />
    <DownloadNuGet OutputFilename="$(NuGetExePath)" Conditi
```

```
<Target Name="RestorePackages" DependsOnTargets="CheckPrer
    <Exec Command="$(RestoreCommand)"
          Condition="'$(OS)' != 'Windows_NT' And Exists('$

    <Exec Command="$(RestoreCommand)"
          LogStandardErrorAsError="true"
          Condition="'$(OS)' == 'Windows_NT' And Exists('$
</Target>

<Target Name="BuildPackage" DependsOnTargets="CheckPrerequ
    <Exec Command="$(BuildCommand)"
          Condition=" '$(OS)' != 'Windows_NT' " />

    <Exec Command="$(BuildCommand)"
          LogStandardErrorAsError="true"
          Condition=" '$(OS)' == 'Windows_NT' " />
</Target>

<UsingTask TaskName="DownloadNuGet" TaskFactory="CodeTaskF
AssemblyFile="$(MSBuildToolsPath)\Microsoft.Build.Tasks.v4
    <ParameterGroup>
        <OutputFilename ParameterType="System.String" Requ
    </ParameterGroup>
    <Task>
        <Reference Include="System.Core" />
        <Using Namespace="System" />
        <Using Namespace="System.IO" />
        <Using Namespace="System.Net" />
        <Using Namespace="Microsoft.Build.Framework" />
        <Using Namespace="Microsoft.Build.Utilities" />
        <Code Type="Fragment" Language="cs">
            <![CDATA[
            try {
                OutputFilename = Path.GetFullPath(OutputFi

                Log.LogMessage("Downloading latest version
                WebClient webClient = new WebClient();
                webClient.DownloadFile("https://nuget.org/

                return true;
            }
            catch (Exception ex) {
                Log.LogErrorFromException(ex);
                return false;
            }
        ]]>
```

```
 <UsingTask TaskName="SetEnvironmentVariable" TaskFactory=
AssemblyFile="$(MSBuildToolsPath)\Microsoft.Build.Tasks.v4
    <ParameterGroup>
        <EnvKey ParameterType="System.String" Required="tr
        <EnvValue ParameterType="System.String" Required="
    </ParameterGroup>
    <Task>
        <Using Namespace="System" />
        <Code Type="Fragment" Language="cs">
            <![CDATA[
            try {
                Environment.SetEnvironmentVariable(EnvKey,
System.EnvironmentVariableTarget.Process);
            }
            catch  {
            }
        ]]>
        </Code>
    </Task>
</UsingTask>
```

answered May 14 '15 at 18:10

Sofia Khwaja

554 3 8 17