Metadata file '.dll' could not be found

Ask Question

555

94

I am working on a WPF, C# 3.0 project, and I get this error:

```
Error 1 Metadata file
'WORK=- \Tools\VersionManagementSystem\BusinessLogicLayer\bin\Debug
\BusinessLogicLayer.dll' could not be found C:\-=WORK=- \Tools
\VersionManagementSystem\VersionManagementSystem\CSC VersionManagementSystem
```

This is how I reference my usercontrols:

```
xmlns:vms="clr-namespace:VersionManagementSystem"
<vms:SignOffProjectListing Margin="5"/>
```

It happens after every failed build. The only way I can get the solution to compile is to comment out all my user controls and re-build the project, and then I uncomment the usercontrols and everything is fine.

I have checked build orders and dependencies configurations.

As you can see, it seems to have truncated the DLL file's absolute path... I have read that there is a bug with the length. Is this a possible problem?

It's very annoying and having to comment, build, and uncomment, the build is becoming extremely tiresome.

c# .net wpf visual-studio-2008 c#-3.0

edited Apr 19 '18 at 14:08

Peter Mortensen

13.7k 19 86 113

asked Sep 14 '09 at 14:19

Oliver
16.3k 7 43 63

2 I had a similar problem (getting the same error that is indicated at the title) and handled it by cleaning and rebuilding the project. To correctly reference other projects, I do have no idea.. – phoad Nov 11 '13 at 3:24

4 I have marked Matt's answer as it seems to have worked for most people however this did not resolve my original issue. I still think it is related to the Windows max path limit. See my answer below. – Oliver Feb 18 '14 at 6:46

1 possible duplicate of Metadata file '...\Release\project.dll' could not be found in Visual Studio – Des Horsley Aug 7 '15 at 3:34

67 Answers

1 2 3 next

695

I just had the same problem. Visual Studio isn't building the project that's being referenced.

1. Right click on the solution and click Properties.
2. Click Configuration on the left.
3. Make sure the check box under "Build" for the project it can't find is checked. If it is already checked, uncheck, hit apply and check the boxes again.

edited Apr 29 '15 at 9:11

krizzzn
1,006 7 11

answered Jul 18 '13 at 12:42

Matt_Bro
8,480 2 20 38

151 And, in my case, even though the check box was checked, unchecking it and checking it again fixed the issue. – ngm Oct 8 '13 at 13:20

10 This fixed my problem - I had to do it for both Release and Debug modes on the solution properties. Thanks! – theJerm Nov 21 '13 at 23:45

15 Holy cow, unchecking and checking it again also worked for me! – Memet Olsen Dec 23 '13 at 11:50

106 Simble uncheck/check didn't solve issue so I've had to do next steps: - clean solution - uncheck all build checkboxes - restart VS - check all build checkboxes - build solution – frankie Apr 8 '14 at 9:54

25 The other thing to do is check each of the project dependencies, for some reason it wasn't setting this automagically. Solution Properties -> Common Properties -> Project Dependencies. – Anicho Apr 15 '14 at 11:05

190

This can still happen in newer versions of Visual Studio (I just had it happen on Visual Studio 2013):

Another thing to try is to close Visual Studio and delete the `.suo` file that is next to the `.sln` file. (It will be re-generated the next time you `Save all` (or exit Visual Studio)).

I've had this problem when adding new projects to the solution on another machine and then pulling the revisions in, but the `.suo` file can be corrupted in other cases as well and lead to very strange Visual Studio behaviour, so deleting it is one of the things I always try.

Note that deleting the `.suo` file will reset the startup project(s) of the solution.

More on the `.suo` file is here.

edited Apr 19 '18 at 14:29

Peter Mortensen
13.7k 19 86 113

answered Apr 22 '14 at 9:39

corvuscorax
4,255 3 24 29

18 This fixed the problem for me. Also worth mentioning is that `.suo` files are hidden. So you'll have to setup your explorer to show hidden files. – George Howarth Jun 20 '14 at 15:09

5 I'm working with Xamarin project and the .suo file is located in the .vs/ folder . I tried deleting it and it didn't solve my problem – user3596965 Feb 27 '16 at 13:43

32 This worked for me too. But in Visual Studio 2015, the `.suo` file is both hidden and sits within a hidden `.vs` directory beside the `.sln` . e.g.: if the solution file is `c:\foo\mysolution.sln` then look for `c:\foo\mysolution\.vs\mysolution\v14\.suo` – Wyck Nov 25 '16 at 18:15

3 For VS2017, for simplicity, I just deleted the `.vs` hidden folder instead which also deleted the `.suo` file. I reopened the solution, fixed one more unrelated error, and the issue was resolved. – user3613932 Aug 2 '18 at 23:22

125

The suggested answer did not work for me. The error is a decoy for another problem.

I found out that I was targeting a slightly different version of .NET and this was flagged as a warning by the compiler, but it was causing building to fail. This should have been flagged as an error and not a warning.

edited Apr 19 '18 at 15:15

Peter Mortensen
13.7k 19 86 113

answered Dec 15 '15 at 21:15

jordan koskei
1,636 1 10 12

9 I was able to fix by matching the framework for the project to the higher version indicated in the warning message by right clicking on the project > Properties > Application > Target Framework. – StronglyTyped Apr 27 '16 at 15:18

1 Same for me, using vs2015. – bruno.bologna Jun 22 '16 at 18:46

87

Well, my answer is not just the summary of all the solutions, but it offers more than that.

Section (1):

In general solutions:

I had four errors of this kind (‘metadata file could not be found’) along with one error saying 'Source File Could Not Be Opened (‘Unspecified error ‘)'.

I tried to get rid of ‘metadata file could not be found’ error. For that, I read many posts, blogs, etc. and found these solutions may be effective (summarizing them over here):

1. Restart Visual Studio and try building again.
2. Go to **'Solution Explorer'**. Right click on Solution. Go to **Properties**. Go to **'Configuration Manager'**. Check if the checkboxes under **'Build'** are checked or not. If any or all of them are unchecked, then check them and try building again.
3. If the above solution(s) do not work, then follow sequence mentioned in step 2 above, and even if all the checkboxes are checked, uncheck them, check again and try to build again.
4. **Build Order and Project Dependencies:**

Go to **'Solution Explorer'**. Right click on Solution. Go to **'Project Dependencies...'**. You will see two tabs: **'Dependencies'** and **'Build Order'**. This build order is the one in which solution builds. Check the project dependencies and the build order to verify if some project (say 'project1') which is dependent on other (say 'project2') is trying to build before that one (project2). This might be the cause for the error.

5. **Check the path of the missing .dll:**

   Check the path of the missing .dll. If the path contains space or any other invalid path character, remove it and try building again.

   If this is the cause, then adjust the build order.

Section (2):

My particular case:

I tried all the steps above with various permutations and combinations with restarting Visual Studio a few times. But, it did not help me.

So, I decided to get rid of other error I was coming across ('Source File Could Not Be Opened (‘Unspecified error ‘)').

I came across a blog post: *TFS Error–Source File Could Not Be Opened (‘Unspecified error ‘)*

I tried the steps mentioned in that blog post, and I got rid of the error **'Source File Could Not Be Opened (‘Unspecified error ‘)'** and surprisingly I got rid of other errors **(‘metadata file could not be found’)** as well.

Section (3):

Moral of the story:

Try all solutions as mentioned in section (1) above (and any other solutions) for getting rid of the error. If nothing works out, as per the blog mentioned in section (2) above, ***delete the entries of all source files which are no longer present in the source control and the file system from your .csproj file***.

edited Apr 19 '18 at 14:35

Peter Mortensen
13.7k 19 86 113

answered May 5 '14 at 14:58

Vikram
2,358 5 28 55

4 My problem was Build Order/Project Dependencies. Removing and adding back references from other projects will correct this (I think) but you can just do it yourself as well. – Nacht Jun 9 '15 at 11:11

4 I faced this problem by down-grading `.NET v4.5` project to `.NET v.4` . – guneysus Jul 8 '16 at 13:58

1 Removing "%" from referenced dll path helped me – Boogier Jan 10 '18 at 14:13

1 The solution in Section 2 worked for me! I had another error and when I fixed that the others magically dissappeared. – Martin Johansson Aug 23 '18 at 6:58

1 Excellent thread - thanks – 765tgs Sep 5 '18 at 13:35

33

In my case it was caused by a .NET Framework version mismatch.

One project was 3.5 and the other referencing project 4.6.1.

edited Apr 19 '18 at 15:18

Peter Mortensen
13.7k 19 86 113

answered Apr 8 '16 at 14:20

Eric Schneider
3,629 2 33 49

1 This also happens between 4.5.2 Vs. 4.6 – AzzamAziz Jul 10 '17 at 19:05

1 Indeed, I had one of 4.6.1 and the rest was 4.5.2, thanks! – Mason Oct 6 '17 at 10:58

6 Yes, seems any time a framework version is different, this happens. Great error Microsoft! – Eric Schneider Oct 6 '17 at 17:11

2 perfect, solved my 4.5.2 -> 4.5.1 problem. – Machado Nov 3 '17 at 20:08

23

Closing and reopening Visual Studio 2013 worked for me!

edited Apr 19 '18 at 14:58

Peter Mortensen
13.7k 19 86 113

answered May 28 '15 at 11:20

Roffers
353 1 5 14

4 Restart VS 2015 also works. – B.Kosmowski Sep 19 '16 at 8:54

16

Well, nothing in the previous answers worked for me, so it got me thinking about why am I clicking and hoping when as developers we should really try to understand what is going on here.

It seemed obvious to me that this incorrect meta data file reference must be held somewhere.

A quick search of the .csproj file showed the guilty lines. I had a section called <itemGroup> that seemed to be hanging onto the old

incorrect filepath.

```
<ItemGroup>
    <ProjectReference
Include="..\..\..\MySiteOld\MySite.Entities\MySite.Entities.csproj">
        <Project>{5b0a347e-cd9a-4746-a3b6-99d6d010a6c2}</Project>
        <Name>Beeyp.Entities</Name>
    </ProjectReference>
...
```

So a simple fix really:

1. Backup your .csproj file.
2. Find the incorrect paths in the .csproj file and rename appropriately.

Please **make sure you backup your old .csproj before you fiddle**.

edited Apr 19 '18 at 14:49

Peter Mortensen
13.7k 19 86 113

answered Nov 27 '14 at 9:22

Alex Stephens
1,595 1 19 25

28 please MAKE SURE YOU USE VERSION CONTROL BEFORE YOU DO ANYTHING – svick Sep 18 '15 at 14:18

13

I also met this problem. Firstly you have to manually build you DLL project, by right-click, Build. Then it will work.

edited Dec 17 '13 at 12:02

BlueRaja - Danny Pflughoeft

58.7k 21 153 241

answered Sep 14 '12 at 20:36

user1678541

13 While this fix works, it doesn't actually fix the problem and could lead to more underlying problems. First of all, if you're working with code in a repository, it is bad form to require a new developer to jump through hoops to get the code to a point where it will build. Second of all, in order to see changes in the referenced project, you would have to manually rebuild it every time. Please see my answer for a more robust fix to the problem. – Matt_Bro Jul 18 '13 at 12:45

13

I got the same error "Metadata file '.dll' could not be found", and I tried several things described above, but the reason for the error was that I was referencing third-party DLL file which was targeting a .NET version higher that my project target .NET version. So the solution was to change the target framework of my project.

edited Apr 19 '18 at 15:16

Peter Mortensen

13.7k 19 86 113

answered Feb 11 '16 at 13:02

Mladen Nikolov

166 1 5

10

In my case, I have my installed directory in mistaken ways.

If your solution path is something like "My Project%2c Very Popular%2c Unit Testing%2c Software and Hardware.zip", it cannot resolve the metadata file, perhaps we should prevent some invalid words like %2c.

Renaming the path into normal name resolved my issue.

answered Mar 27 '15 at 8:03

masphei

101 2 5

1 Could you elaborate more your answer adding a little more description about the solution you provide? – abarisone Mar 27 '15 at 8:27

10

I added a new project to my solution and started getting this.

The reason? The project I brought in was targeting a different .NET framework (4.6 and my other two were 4.5.2).

edited Apr 19 '18 at 15:13

Peter Mortensen

13.7k 19 86 113

answered Dec 11 '15 at 16:18

Todd Vance

2,308 5 34 63

1 I dont know why now but i was running my projects for a year like that. my sub project was 4.6.1 and main project was 4.5.2. it worked without any problem. suddenly i am getting this error but i dont want to downgrade the sub project because it has feature that exist in 4.6.1 i dont believe this is the problem. Microsoft explains it should still be working – batmaci Oct 24 '16 at 10:21

9

For me it occurred when I included a new project to a solution.

Visual Studio automatically selects .NET framework 4.5.

I changed to version .NET 4.5.2 like the other libraries, and it worked.

edited Apr 19 '18 at 15:41

Peter Mortensen
13.7k 19 86 113

answered Mar 31 '17 at 13:02

Andre Mesquita
427 4 15

8

For me, it was trying to find a DLL in a path that used to contain the Project, but we'd moved it to a new directory. The Solution had the correct path to the Project, but Visual Studio somehow kept looking in the old location.

Solution: Rename each problem Project - just add a character or whatever - then rename it back to its original name.

This must reset some global cache of some kind in Visual Studio, because this clears both this issue up and several like it, while things like Clean do not.

answered Jan 28 '14 at 6:14

Chris Moschini
26.4k 14 123 159

8

For me the following steps worked:

- Find the project that is not building
- Remove/add references to projects within the solution.

edited Apr 19 '18 at 14:37

Peter Mortensen

13.7k 19 86 113

answered Jul 11 '14 at 13:54

Baglay Vyacheslav

81 1 2

8

I was pulling my hair out with this problem also, but after trying the previous answers the only thing that worked for me was to open each project in my solution 1 by 1 and build them individually.

Then I closed Visual Studio 2013, reopened my solution and it compiled fine.

It's strange, because if I clicked each project in my Solution Explorer and tried to build them that way, they all failed. I had to open them alone in their own solutions.

edited Apr 19 '18 at 14:54

Peter Mortensen

13.7k 19 86 113

answered Dec 19 '14 at 20:35

prospector

2,202 1 15 34

1 Ugh, this. So many things Microsoft require a restart to work again. – Yatrix Jul 5 '16 at 16:57

7

My instance of the problem was caused by a common project that had a duplicate class name in it (under a different filename). It is strange that Visual Studio could not detect that and instead just blew up the build process.

edited Apr 19 '18 at 14:38

Peter Mortensen
13.7k 19 86 113

answered Jul 24 '14 at 19:12

Eric
81 1 1

7 It is an additional solution to the same problem. I know the OP is old, but based on the last couple of posts, people are still finding other causes. Just trying to save the next guy some frustration as none of the other solutions worked for me either. – Eric Jul 24 '14 at 19:22

4 I'm not critiquing anyone's response, just offering an alternate solution to the same symptom. – Eric Jul 24 '14 at 20:25

7

I got this problem in Visual Studio 2012 in a solution that had many projects. Rebuilding each project in the solution manually in the same order as the Project Build Order (right-click and rebuild in Solution Explorer) fixed it for me.

Eventually I got to one that gave me a compile error. I fixed the error, and the solution would build correctly after that.

edited Apr 19 '18 at 14:43

Peter Mortensen
13.7k 19 86 113

answered Oct 31 '14 at 0:33

dan-gph
10.2k 8 46 73

6

In my case, the problem was caused by a simple build error,

> error CS0067: The event 'XYZ' is never used

that, for any reason, did not show up in the error window.

Because of that, the Visual Studio build system seemed to miss the error and tried to build dependent projects, which in turn failed with the annoying metadata message.

The recommendation is -as stupid as it may sound-:

First look at your *Output Window*!

It took me half an hour before this idea hit me...

edited Apr 19 '18 at 15:10

Peter Mortensen
13.7k 19 86 113

answered Nov 1 '15 at 9:21

Heinz Kessler
844 7 12

5

In my case the issue was that I'd manually deleted a non-compilation file which was marked as "missing". Once I deleted the reference to the now-missing file and recompiled - all was well.

answered Apr 10 '14 at 13:42

David Ford
326 1 4 9

5

Coming back to this a few years later, this problem is more than likely related to the Windows maximum path limit:

Naming Files, Paths, and Namespaces, *Maximum Path Length Limitation*

edited Apr 19 '18 at 14:18


Peter Mortensen
13.7k 19 86 113

answered Feb 18 '14 at 6:47


Oliver
16.3k 7 43 63

5

I too had the same error. It hides as in the below path. The path which I referred to for the DLL file is like "D:\Assemblies Folder\Assembly1.dll".

But the original path in which the assembly referred was "D:\Assemblies%20Folder\Assembly1.dll".

Due to this path name variation, the assembly could not be retrieved from its original path and hence throws the "Metadata not found" error.

The solution is in Stack Overflow question *How do I replace all the spaces with %20 in C#?*.

edited Apr 19 '18 at 15:21


Peter Mortensen
13.7k 19 86 113

answered Jun 18 '16 at 5:49


Arun Prasad
123 1 13

5

It looks like such kind of errors related to the fact that Visual Studio doesn't provide correct information about an error. The developer doesn't even understand the reason for the failed build. It can be a syntax error or something else. In common, to solve such problems you should find the root of the problem (for example, look at the build log).

In my case the problem was in fact that the `Error List` window didn't show any errors. But really there were syntax errors; I found these errors in the `Output` window, and after fixing them, the problem was solved.

edited Apr 19 '18 at 15:38

Peter Mortensen
13.7k 19 86 113

answered Jan 31 '17 at 9:22

burzhuy
785 7 23

5

I'd faced the same problem. In my case I'd referenced to a class library project with higher **.Net version** than my project and VS failed to build the project and raised the same error you posted.

I simply set **.Net version** of my class library project(the one that had broken the build) identical to the .Net version of referenced project and problem solved.

answered Jun 21 '18 at 16:07

Code_Worm
846 1 14 20

1 This!!! While the above answer was good this was something i just completely overlooked. Thank you good sir. – Rhys Johns Aug 8 '18 at 4:24

4

If you have a space in your solution name, this will also cause the issue. Removing the space from your solution name, so path doesn't contain %20 will solve this.

answered Jun 17 '15 at 17:24

Ajaco
149 8

4

Just pointing out the blatantly obvious: if you don't have "Show output window when build starts" enabled, make sure you're noticing if your build is failing (small "build failed" error in lower left)!!!!

answered Jul 9 '15 at 17:42

tbone
2,992 15 71 116

4

I had this error when I was trying to publish a web application. Turned out that one of a class properties was wrapped into

```
#if DEBUG
    public int SomeProperty { get; set; }
#endif
```

but the property usage was not. The publishing was done in Release configuration without the `DEBUG` symbol, obviously.

answered Sep 4 '15 at 9:31

Dmitri Trofimov
489 2 17

4

This error may be shown if you use fake assemblies. Removing fakes leads to successful build of the project.

edited Apr 19 '18 at 14:50

Peter Mortensen
13.7k 19 86 113

answered Dec 5 '14 at 13:14

FLCL

1,442 1 13 37

4

I am running Visual Studio 2013.

It appears that the build dependencies were incorrect. Deleting the *.suo files did fix the problems I had.

edited Apr 19 '18 at 14:58

Peter Mortensen

13.7k 19 86 113

answered Apr 9 '15 at 20:03

DrBB

43 5

4

For my case it was that I had commented out classes in a specific (empty) namespace:

```
namespace X.Y.Z.W
{

    // Class code

}
```

When I removed the namespace code and the import (using) commands of it - it fixed the problem.

In the build it was also saying - along with the missing DLL file of the project:

> error CS0234: The type or namespace name 'W' does not exist in the namespace 'X.Y.Z' (are you missing an assembly reference?)

edited Apr 19 '18 at 15:12

Peter Mortensen
13.7k 19 86 113

answered Nov 24 '15 at 15:30

Michail Michailidis
4,800 4 32 64

3

I had the same issue. In my case, the project would still build in release mode and it was just when I tried to build in debug that it failed.

What I ended up doing to fix the issue was simply copy all of the dlls (and other files from my release folder) into my debug folder. After doing this for every project, the errors melted away.

answered May 13 '15 at 18:01

Jack Fairfield
827 11 19

---

1 2 3 next

protected by Community ♦ Jul 9 '15 at 4:54

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?