Targeting multiple versions of .net framework

Asked 16 years, 3 months ago Modified 5 years, 10 months ago

Viewed 4k times

4

Suppose I have some code that would, in theory, compile against *any* version of the .net framework. Think "Hello World", if you like.

If I actually compile the code, though, I'll get an executable that runs against one *particular* version.

Is there any way to arrange things so that the compiled exe will just run against whatever version it finds? I strongly suspect that the answer is no, but I'd be happy to be proven wrong...

Edit: Well, I'll go to the foot of our stairs. I had no idea that later frameworks would happily run exe's compiled under earlier versions. Thanks for all the responses!

.net compilation version

Share

Improve this question

Follow

edited Jan 26, 2019 at 19:45

Javier Enciso

55 ● 1 ● 2 ● 11

asked Sep 4, 2008 at 15:00

Matt Bishop

1,487 ● 1 ● 12 ● 19

7 Answers

Sorted by: Highest score (default)

6

Im not sure if this is correct, but i'd try to compile it for the lowest version, the higher versions should be able to run the lower versions exe's.

Share Improve this answer

Follow

answered Sep 4, 2008 at 15:03

John Boker

83.7k ● 17 ● 100 ● 130

What if I wanted to take advantage of some features of the newer framework versions but still have the program work on the older ones? What is the correct approach in such case? – julealgon Mar 10, 2014 at 22:12

1 This may not always be the case. I built a Windows Service targeted for .NET 2.0 for the same reason, however Windows 8.0+ does not have 2.0-3.5 installed by default (needs to be activated from 'Add or Remove Features'). – Trent Seed Dec 4, 2014 at 1:08

Read ScuttGu's post about VS 2008 Multi-Targeting Support

3

> One of the big changes we are making starting with the VS 2008 release is to support what we call "Multi-Targeting" - which means that Visual Studio will now support targeting multiple versions of the .NET Framework, and developers will be able to start taking advantage of the new features Visual Studio provides without having to always upgrade their existing projects and deployed applications to use a new version of the .NET Framework library.
>
> Now when you open an existing project or create a new one with VS 2008, you can pick which version of the .NET Framework to work with - and the IDE will update its compilers and feature-set to match this. Among other things, this means that features, controls, projects, item-templates, and assembly references that don't work with that version of the framework will be hidden, and when you build your application you'll be able to take the compiled output and copy it onto a machine that only has an older version of the .NET Framework installed, and you'll know that the application will work.

That way you can use VS2008 to develop .NET 2.0 projects that will work on both .NET 2.0, 3.0 and 3.5

Share Improve this answer Follow

answered Sep 4, 2008 at 15:03

Espo
41.9k • 21 • 136 • 161

0

Along side multi targeting, the frameworks are backwards compatible, so something compiled to 1.0 will run on 1.1 and 2. Somthing compiled on 1.1 will run on 2 ... etc.

Share Improve this answer

Follow

answered Sep 4, 2008 at 15:04

DAC
1,789 • 2 • 21 • 32

0

I know @John Boker is correct when it comes to .Net class libraries. You can compile a class library against .Net 1.1 and then use it in a .Net 2.0 or higher project.

I suspect the same is also true for executables.

Share Improve this answer

Follow

edited May 23, 2017 at 12:13

Community Bot
1 • 1

answered Sep 4, 2008 at 15:05

samjudson
56.8k • 7 • 60 • 69

0

with 2005 & 2008, yes (on CLR 2.0)

With 2003, no.. because it compiles down to CLR 1.1

You could theorectically write some code using #if (DOTNET35) and such so that you don't use features outside the compilers knowledge and then run the desired compiler on the app... I question the usefulness of this though.

Share Improve this answer

Follow

answered Sep 4, 2008 at 15:05

Ben Scheirman

40.9k ● 21 ● 103 ● 139

0

Well, AFAIK, all .NET versions (except version 1.x) compile to the same bytecode. In case of C#, all new features are simply syntactic sugar, which get transformed into C# 2.0 constructs when compiling. The key point where things could go wrong is when you use C# 3.0 or 3.5 specific DLLs. They don't work well with the .NET 2.0 framework, so you can't use those.

I can't really think of a workaround for this, sorry :(

Share Improve this answer

Follow

answered Sep 4, 2008 at 15:06

Erik van Brakel

23.8k ● 3 ● 53 ● 66

0

On the subject of which .NET framework the user has installed, there is also a new option with the Client Profile that’s available with .NET 3.5 SP1. This basically allows you to ship a small (277k) bootstrap program which downloads and installs the required files (A subset od the full .NET framework).

For more information, and general tips on creating a small .NET installation, see this great blog entry by Scott Hanselman.

Share Improve this answer

Follow

answered Sep 4, 2008 at 15:55

Morten Christiansen

19.5k ● 23 ● 73 ● 96