INFO: x86 vs x64 vs AnyCPU

Covered in this Article

Managed / Unmanaged code and "Bitness"

How "AnyCPU" works

Web Applications and Services vs. Executable (WinForms, WPF, Console, etc...)

Quick Summary

This article will attempt to cover targeting AnyCPU with DotImage components which is a

You must target either x86 or x64 as our components "Have bitness". Visual Studio / .NET does not allow you to mix "bitness" in the same project, and as soon as you need to reference a non-AnyCPU DLL (which most of the key functionality of DotImage lives in), you are committed to that "bitness" for the project.

If you reference our x86 or x64 DLLs in a project that targets "AnyCPU" you will find that it fails under many/most situations. We ran into issues with older versions of Visual Studio defaulting to AnyCPU... since Visual Studio itself is 32 bit, projects targeting AnyCPU but using our x86 components appeared to work when run in Visual studio host process and/or when run on 32 bit machines, but would immediately crash when attempting to run in a 64 bit context.

So, target x86 or x64. Since x86 apps run on both x64 and x86 systems, the most compatible choice is to build x86. If you MUST build a 64 bit solution, you'll need to target x64 and use our x64 DLLs.

Helpful Links for x64-related issues

INFO: Warning: Referenced Assembly Targets a Different Processor

Detailed Explanation

Visual Studio provides several "platform target" options:

INFO: x86 vs x64 vs AnyCPU

- AnyCPU
- x86
- x64
- Arm64

Since DotImage 6.0, Atalasoft has supplied DLLs for x86 and x64, and since DotImage 9.0, we've supplied two sets - one labeled 2.0 (for .NET framework 2.0, 3.0, and 3.5) and 4.0 (for .NET framework 4.0 and 4.5)

Later DotImage versions moved .NET Framework to 4.5.2 and eventually to 4.6.2 - generally following the "what was the current oldest supported .NET Framework 4.x from MA at the time of the major new release"

We have never had any support for Itanium, and currently do not have any support for Arm64.

DotImage 10.0 introduced AnyCPU support for an extremely limited subset of DLLs that are specific to certain SharePoint use cases, but the AnyCPU DLLs found in

C:\Program Files (x86)\Atalasoft\DotImage 10.X\bin\4.0\AnyCPU\

Should not be taken to mean that you are free to target AnyCPU with your applications.

This is because we simply do not have AnyCPU versions of any of our DLLs except

- Atalasoft.dotImage.dll
- Atalasoft.dotImage.WebControls.dll
- Atalasoft.Shared.dll

and those are for specific SharePoint WebPart use... they contain dependencies that will clash with non-SharePoint solutions.

As soon as you need components that exist in any of the others: Atalasoft.dotImage.Lib.dll or Atalasoft.dotImage.WinControls.dll, or any other of our components, there is no AnyCPU version ... they "have bitness"

Why No True AnyCPU Support

INFO: x86 vs x64 vs AnyCPU

"OK then, Why Don't You Have Full AnyCPU Support in DotImage?"

Part of it has to do with memory management... a 64 bit address space versus a 32 bit address space... when you have image processing, you need to put images in to contiguous blocks of memory (a requirement from Windows) and when you do many image processing operations, you're doing pointer arithmetic... this is done in unmanaged code and "has bitness"

Yes, in theory, we could write an abstraction layer to adjust dynamically between 32 bit addressing and 64 bit addressing but it would be at the cost of orders of magnitude of slowdown for processing unless it was written from the ground up... and there are hundreds of thousands of lines of code in DotImage...

Simply put, it is a not likely that we will be able to provide full AnyCPU support in DotImage.

This is not a matter that we can put in a feature or bug request for... this is not a case of us simply choosing to do or not do something because one way was easy or hard.... the practical upshot is how AnyCPU actually works.

When an AnyCPU application is run, it needs to determine whether it's going to run 32 bit or 64 bit based on execution context.

Managed code (C#, VB.NET, etc...) can live in that world, but the basic building blocks that power DotImage are in unmanaged code (C/C++) and have been years in the making, so converting them to Managed code without sacrificing performance and without potentially breaking thousands of things is a long term (years) project.

The closest you can come would be to build both an x86 and an x64 version of your application, and then have some controller/wrapper application determine on the fly which one to run. If you target AnyCPU with Atalasoft components, what will happen is that depending on whether you're using our x86 or x64 DLLs, it will only work when the AnyCPU app runs under the matching context:

- Run on a 32 bit machine, and it will only work if the Atalasoft DLLs are the x86 ones
- Run on a 64 bit machine, and it will only work if the Atalasoft DLLs are the x64 ones

The process of properly embedding licensing is just about impossible due to the fact that

visual studio itself is a 32 bit app (x86 only) and the license compiler (lc.exe) "has bitness" if you're running the lc.exe in x86, it won't be able to "see" the x64 DLLs for DotImage and vice versa (if you force it to x64, it won't "see" the x86 DLLs)

Wrapping Up

We'd very much like to be able to provide full AnyCPU support, but for the foreseeable future, you will need to pick x86 or x64 for your project(s) that use DotImage, and match that "bitness" with the correct version of the Atalasoft DLLs.

See Also

INFO: Bitness Roundup Whitepaper: x86, x64, AnyCPU and ARM

ERRMSG: Cause of System.BadImageFormatException, "An attempt was made to load a program with an incorrect format"

FIX: Drag-Drop of Some Atalasoft Viewers Not Adding Control to Designer

INFO: Workaround for a License Compiler Exception When Targeting x64 in .NET Framework 4.0 and up

INFO: Workaround for a license compiler exception when targeting x64 in .NET framework 2.0, 3.0, or 3.5

INFO: Warning: Referenced assembly targets a different processor

Original Article

Q10401 - INFO: x86 vs x64 vs AnyCPU

Last Update

2026-10-12 - TD - (Fixing formatting issues)