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

When deploying an application, you might run into this error:

```
ystem.BadImageFormatException: Could not load file or assembly
'SOME_ATALASOFT_ASSEMBLY_NAME_HERE' or one of its dependencies.

An attempt was made to load a program with an incorrect format.
```

The exact DLL names may differ, but in this case, it is most important to note "BadImageFormat" is referring to the format of the DLL, and NOT anything to do with imaging formats as in jpeg, tiff, png, bmp, etc...

```
An attempt was made to load a program with an incorrect format"
```

This is the last part of the exception message that quite frequently plagues developers attempting to get our sample applications/demos up and running. And even when building your own app from scratch, this can be a vague, and annoying error.

**NOTE:** If the message ends with "Specified module could not be found" instead of the message about incorrect format, see Q10318 - INFO: .NET 4.0 Deployment Errors

The root cause is that this message means there's a "bitness mismatch" somewhere.. a 32-bit process (x86) is attempting to call a 64 bit DLL, or a 64 bit process is attempting to call in a 32 bit DLL. Typically this will happen on a 64-bit machine when building targeting AnyCPU. AnyCPU marks your output assembly/executable as being able to run under either x86/x64 and it is determined which at runtime. If the assemblies it finds at runtime don't match the CPU target (x86/x64) it is running in it fails with that error

Atalasoft components do not support "AnyCPU" so if you build an AnyCPU app then you have the problem that on 32 bit machines, it will need our 32 bit (x86) dlls but on 64 bit it will need x64

## Desktop (winforms/WPF/Console) Applications

In your visual studio project, you need to pick either x86 or x64.. Visual Studio being 32-bit (x86) it's recommended to go with x86 unless you have a compelling business reason to make an x64 application)

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

Go to the project properties. In C#, look in the BUILD tab, in VB look in the COMPILE tab (possibly under Advanced compile options depending on version) and find the entry for Platform Target

If it's AnyCPU, set it to x86

If you are set to x64 and don't need x64 change it to x86

Once you've chosen your 'Bitness" for compiling, you will now need to triple check that you are referencing the correct 'bitness' for your Atalasoft DLLs

Look at your application's references and ensure that the Atalasoft DLLs are either not showing x64 (x86 DLLs) if you';re targeting x86 or if you are targeting x64, make sure you're referencing x64 versions

We ship 4 sets of DLLs with modern DotImage

in 11.0 and newer its

**.NET 3.5 DLLs (for .NET 3.5 only)**

C:\Program Files (x86)\Atalasoft\DotImage 11.1\bin\3.5\x64\

C:\Program Files (x86)\Atalasoft\DotImage 11.1\bin\3.5\x86\

**.NET 4.5.2 DLLs (for 4.5.2 and higher only)**

C:\Program Files (x86)\Atalasoft\DotImage 11.1\bin\4.5.2\x64\

C:\Program Files (x86)\Atalasoft\DotImage 11.1\bin\4.5.2\x86\

For 10.7 and older it would be

**.NET 2.0 DLLs (for 3.0 and 3.5 - limited 2.0 support)**

C:\Program Files (x86)\Atalasoft\DotImage 10.7\bin\2.0\x64\

C:\Program Files (x86)\Atalasoft\DotImage 10.7\bin\2.0\x86\

**.NET 4.0 DLLS (.NET 4.0 and up)**

C:\Program Files (x86)\Atalasoft\DotImage 10.7\bin\4.0\x64\

C:\Program Files (x86)\Atalasoft\DotImage 10.7\bin\4.0\x86\

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

Make sure you reference DLLs from the correct "biness" and framework for what you're targeting

and rebuild.

If you're building x64 apps, you'll possibly need to take additional steps depending on your licensing needs

INFO: Workaround for a license compiler exception on 64-bit systems with VS2008/VS2010 .Net 2.0

INFO: Workaround for a license compiler exception on 64-bit systems with VS2010 .Net 4.0

## Web Applications in Visual Studio Application Development Server (IISExpress)

Microsoft Visual Studio has a built in ASP.NET Development Web Server. In older versions, it was called Cassini. In modern versions (since VS2012) it's called IIS Express.

Visual Studio has been (and remains) a 32 bit (x86) process. This can cause some issues as it is used to build solutions in both x86 and in x64. One area where this has been especially difficult has been in the built in web server which was x86 by default and very difficult to change to x64 even though the vast majority of web servers in production (including MS's IIS) have been running x64 fully for years.

This leads to the problem of .NET web developers sometimes finding that they've been developing in x86 locally and having to face issues of fixing the "bitness" when deploying to IIS.

Starting in Visual Studio 2013, MS added an easy switch for being able to select between x64 and x86 for IIS Express, and also changed the default to x64.

Since Atalasoft components "have bitness" this switch can and has caused issues as some of our older example apps were targeting x86 (to work with the most common configurations of IIS Express)

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

SO, here's how:

### VS2013 and newer

Tools->Options->Projects and Solutions->Web Projects

check the Use 64-bit version of IISExpress for web sites and projects

### VS2012 and older

If you are working on ASP.NET MVC web sites in Visual Studio 2013 (VS2013), you need to make one registry change if you want to run IIS Express as a 64-bit process by default. Use one of the methods, below.

### Command-Line:

reg add HKEY_CURRENT_USER\Software\Microsoft\VisualStudio\11.0\WebProjects /v Use64BitIISExpress /t REG_DWORD /d 1

### Regedit:

1. Navigate to: HKEY_CURRENT_USER\Software\Microsoft\VisualStudio\11.0\WebProjects
2. Make sure there is a REG_DWORD value named “Use64BitIISExpress”. If not, create it.
3. Set its value from 1 (default = 0).

That’s it. Now you can debug your 64-bit web sites.

If you don't do this, you will likely get a message similar to:

"Could not load file or assembly or one of its dependencies. An attempt was made to load a program with an incorrect format."

Original source of this info: I found this solution for Visual Studio 2012 (VS2012) on Stack Overflow here.

## IIS (local instance, or production server)

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

Microsoft IIS web server control over "bitness" happens in Advanced App Pool settings

There's a checkbox for "enable 32-bit applications"

If it is checked, then the app pool is running 32 bit (x86) if it is unchecked, IIS runs in 64 bit (the default)

For more information, please see:

INFO: Bitness Roundup Whitepaper: x86, x64, AnyCPU

Original Article:

Q10165 - ERRMSG: Cause of System.BadImageFormatException, "An attempt was made to load a program with an incorrect format"

Atalasoft Knowledge Base
https://www.atalasoft.com/kb2/KB/50288/ERRMSG-Cause-of-SystemBadImageFormat...