Is it worth learning to use MSBuild? [closed]

Asked 16 years, 3 months ago Modified 1 year, 7 months ago

Viewed 4k times

20

Closed. This question is opinion-based. It is not currently accepting answers.

Want to improve this question? Update the question so it can be answered with facts and citations by editing this post.

Closed last year.

Improve this question

I simply wondered whether people thought it was worth learning to use the MSBuild syntax in order to customise the build process for a .net project, or whether it is really not worth it given the ease with which one can build a project using visual studio.

I am thinking in terms of nightly builds, etc., but then couldn't I use a scheduled event which uses the command-line build option built into VS? Are there superior tools out there?

.net msbuild

Share

Improve this question

Follow

asked Sep 6, 2008 at 22:26

ljs

37.8k • 36 • 109 • 124

10 Answers

Sorted by: Highest score (default)

15

I would say YES.
The neat thing about MSBuild is that if you modify your csproj files to include custom build steps then those steps will happen from within VS or from MSBuild. Also if you ever have a build server you will not need to install full VS, only the SDK to build your projects.

Share Improve this answer

Follow

edited Apr 27, 2023 at 18:59

TylerH

21.2k • 76 • 79 • 110

answered Sep 6, 2008 at 22:29

Andrew Burns

14.4k • 9 • 41 • 42

13

MSBuild is absolutely worth the time to learn. After the initial learning curve (which might be very steep actually) it becomes fairly easy to do the most common build automation steps.

- building assemblies in RELEASE mode
- signing assemblies with strong name
- running unit tests
- modifying xml files / Web.config-s on the fly
- modifying the version number of the assemblies
- validating FxCop / StyleCop etc...
- automated deployment - create SQL databases, IIS websites, windows services etc...

Share Improve this answer Follow

edited Sep 22, 2008 at 21:56

answered Sep 22, 2008 at 21:50

Jivko Petiov

586 ● 7 ● 15

8

MSBuild is definitely worth learning for anyone and everyone writing .NET software. The reason a build server for .NET apps no longer requires Visual Studio to be installed (as Andrew Burns mentioned) is because MSBuild is part of the .NET Framework now.

Knowing MSBuild will give you significant flexibility in choosing what technologies you use to implement continuous integration. Because I took the time to learn MSBuild, I was able to change the CI system one of our teams was using from CruiseControl.NET to TeamCity without much difficulty. Those CI servers, or something like FinalBuilder (which I'm not familiar with), are better options for performing nightly builds than a scheduled task. Learning how to implement custom MSBuild tasks will give you even more flexibility in implementing custom builds. Jivko Petiov listed a number of tasks that MSBuild makes easier. In the case of database deployment and configuration, I've written scripts that do this in MSBuild, and it makes the development and testing process much easier.

If Visual Studio Team System is in your future, applications built using MSBuild will be much easier to move into that environment than those built via alternative means.

There are a lot of resources available to help you get started with MSBuild. I'd start with Inside the Microsoft Build Engine. One of the co-authors also has a ton of stuff on the web, including this site, and a project on CodePlex.

answered Feb 1, 2010 at 18:58

Scott Lawrence

7,243 ● 13 ● 47 ● 64

It sounds like you are a single developer working on your own site. If this is the case, it's not necessary at all, but it is still a good idea for you to learn as a part of your professional experience.

Automated building of projects becomes more necessary as the number of developers working on a project increase. It is very easy for two developers to write incompatible code which will break when it is combined (imagine I'm calling a function foo(int x), and you change the signature to be foo(int x, int y): when we combine our code bases, the code will break.

These types of errors increase in complexity and hassle with the amount of time between integration builds. By setting up nightly builds, or even builds that occur every check-in, these problems are greatly reduced. This practice is pretty much industry standard across projects with multiple developers.

So now, to answer your question: this is a skill that will span across projects and companies. You should learn it to broaden your knowledge and skills as a developer, and to add an important line on your resume.

answered Sep 6, 2008 at 23:36

tsimon

8,010 ● 2 ● 33 ● 44

5

Well, MSBuild is built in, so if you are doing something simple, then yes, it is recommended.

But for something like nightly builds, I would suggest FinalBuilder.

See this question on Build/Configuration Management Tools.

Share Improve this answer

Follow

edited May 23, 2017 at 12:01

Community Bot

1 • 1

answered Sep 6, 2008 at 22:53

Michael Pryor

25.3k • 18 • 73 • 92

2

MSBuild is incredibly simple to use, you can use VS to manage the projects and solution files and just pass the SLN to MSBuild.

Share Improve this answer

Follow

answered Sep 6, 2008 at 23:34

FlySwat

175k • 75 • 248 • 314

2

In a scenario such as yours, where you do not already have a build system, then yes, MSBuild is absolutely worth it. Not only can you use it for a variety of pre-build and post-build tasks (see Jicko Petiov's answer), but you

can also integrate it nicely into a continuous integration environment (such as CruiseControl).

One scenario where it might not be worth it is when you already have an automated/scripted build system in place. For example, I myself haven't taken the time with MSBuild because I've been using NAnt for this task since before MSBuild existed ...

Share Improve this answer Follow

answered Sep 22, 2008 at 21:56

John Rudy
37.8k ● 14 ● 66 ● 101

1

Building from the command line with MSBuild is relatively easy to learn. Start by opening a Visual Studio Command Prompt, and running `msbuild /?`. Just read through the help once and then decide later if you want to learn more details.

Writing project files is a bit more complicated. Most people don't need to learn it, because you can do most things in Visual Studio. However, it's also quite powerful for certain problems.

I have in the past used MSBuild as scripting language, combined with lots of custom tasks. MSBuild has fantastic logging support + built-in dependency management. However, it's not an easy language to learn. PowerShell is a much better choice.

Share Improve this answer

answered Sep 20, 2008 at 16:42

Follow

Jay Bazuzi
46.4k ● 16 ● 115 ● 170

1

If you develop in .net workshop, It does **worth** learning. I have integrated our build process with Jenkins - originally Hudson. As it mentioned above, MSbuild has steep learning curve.However once you grasp the fundamentals, you can start customizing the build. my impression so far - could be naive, bulk of the script is consisted of

```
<PropertyGroup>
   <PropertyKey>value</PropertyKey>
</PropertyGroup>
<ItemGroup>
   <ItemListKey>List values<ItemListKey>
</ItemGroup>
<Task Source="" Target="" />
```

```
    Besides using for build, I successfully used
MSBuild to create a module that manages
    configuration files such as web.config and
foo.exe.config files.
    it is a hybrid module that consists of .net
console app, MSBuild script and batch file.
    what this module does is that during a project
upgrade, it will create a XML transform
    template with connection strings, endpoints
and appSettings from old configuration
    files.
    after the project has been upgraded, the
module will transform newly deployed
    configuration
    files without affecting any new entries. if
you have dozens of configuration files this
    is very effective.
```

Share Improve this answer

Follow

edited May 8, 2014 at 20:02

answered Jan 29, 2013 at 17:57

yantaq

4,028 ● 2 ● 36 ● 34

0

> @kronoz I would say YES. The neat thing about MSBuild is that if you modify your csproj files to include custom build steps then those steps will happen from within VS or from MSBuild. Also if you ever have a build server you will not need to install full VS, only the SDK to build your projects.

==> This is not entirely true. For example, building a setup project on a build server will require Visual studio to be installed!!

Share Improve this answer

Follow

answered Mar 2, 2010 at 12:12

JoDG

1,346 ● 8 ● 18