specify project file of a solution using msbuild

```
msbuild test.sln /t:project /p:Configuration="Release" /p:Platform="x86" /p:BuildProjectReferences=
```

Notice that what is assigned to `/t` is the project name in the solution, it can be different from the project file name.

Also, as stated in How to: Build specific targets in solutions by using MSBuild.exe:

> If the project name contains any of the characters `%`, `$`, `@`, `;`, `.`, `(`, `)`, or `'`, replace them with an `_` in the specified target name.

You can also build multiple projects at once:

```
msbuild test.sln /t:project;project2 /p:Configuration="Release" /p:Platform="x86" /p:BuildProjectRe
```

To rebuild or clean, change `/t:project` to `/t:project:clean` or `/t:project:rebuild`

MSBuild actually works through the use of projects not the solution. The solution is only used to parse it into a temporary project file in MSBuild internally. You should be able to just build the project of interest directly through MSBuild by executing the following command.

```
"msbuild testproject /p:Configuration=Release /p:Platform=x86"
```

There is one major issue I know you could run into using the project directly instead of the solution: if you use the solution to express dependencies between the projects, instead of adding the references to the project and letting the build system work out the dependencies automatically.

If you are enforcing a build order using the sln file, I recommend working those dependencies directly into the proj files and removing them from the sln. This will allow you to invoke any proj file from MSBuild directly and the projects will all build independently without any additional work. You really should treat the sln file as a group of projects to make working in Visual Studio easier and not as a build input.

Posting as information to future seekers

Add the following to the build script and run it once. This will generate the exact targets and other information that msbuild will actually use.

Ex: If you have `.` in the project name or folders msbuild will expect `_` in place of the `.`.

```
set MSBuildEmitSolution=1
```

After getting the information update the build script with the required details.

Tags: **Build** / **Tfs** / **Msbuild** / **Release** / **Devenv**

Related

Will dual Xeons improve Android Studio build times?

Schedule a job once every day on work days in jenkins

Could not find a configuration file for package "ECM" that is compatible with requested version 1.5.0

How to get Colored Build Output from Make in Sublime Text 3?

Visual Studio not auto-building when I press the debug button

Arduino command line vs. Arduino builder

Removing sources after building from them

FFmpeg package for Apple Silicon

flutter release apk error :Execution failed for task ':app:lintVitalRelease'

Visual Studio - Debug Executable Specified in the Debug Profile does not Exist

Recent Posts

Pandas how to find column contains a certain value

Recommended way to install multiple Python versions on Ubuntu 20.04

Build super fast web scraper with Python x100 than BeautifulSoup