# MSBuild调研

## MSBuild是什么？

MSBuild是Microsoft和Visual Studio的生成平台。MSBuild为项目文件提供了一个XML架构，用于控制生成平台处理和生成软件的方式，简单的说就是生成.exe文件。

在命令行中遇到“MSBuild不是内部或外部命令，也不是可运行的程序或批处理文件”时，需要配置环境变量C:\Windows\Microsoft.NET\Framework\v4.0.30319和C:\Windows\Microsoft.NET\Framework64\v4.0.30319（不需要了！！！！），最后在命令行打入MSBuild -v即可看到版本号

## **MSBuild基本概念**

MSBuild有四个基本部分（属性、项、任务、目标）：

MSBuild属性: 属性是一些键/值对，主要用来存储一些配置信息。

MSBuild 项: 主要是存储一些项目文件信息，以及文件的元数据信息（如版本号）。

MSBuild任务: Build过程中的一些原子操作（如CSC、MakeDir）

MSBuild目标: 按特定的顺序将任务组织在一起，并允许在命令行单独指定各个部分。

<https://www.shuzhiduo.com/A/gGdXMGDWz4/>

项目

项目是MSBuild构建文件的顶级节点。

1. <Project>
2. </Project>

**属性**

用来为MSBuild构建过程传递的参数，有多种方式可以定义属性。

1. <PropertyGroup>
2. <Configuration>Release</Configuration>
3. </PropertyGroup>

这样就传入了一个名为Configuration的属性，它的值是Release，PropertyGroup就是专门用来编写属性的组。

**项**

项就是MSBuild构建过程中要用的集合对象了，你可以利用项来在MSBuild中定义你想使用的东西。

1. <ItemGroup>
2. <Foo Include="hello" />
3. </ItemGroup>

这样就定义了一个叫做Foo的项，它包含了一个hello。其中，ItemGroup 是专门用来编写项的组。

**任务**

任务是MSBuild真正要执行的东西，例如编译、打包和下载文件等等任务

1. <Message Text="Hello" />

打印一个hello

**目标**

目标是一组任务的集合，我们简单理解为：为了完成一个目标，需要执行一系列的任务。

1. <Target Name="Print">
2. <Message Text="Hello" />
3. </Target>

定义了一个叫做Print的目标，它用来输出一个Hello。

Project元素还有一个可选特性InitialTargets，也支持多个目标。如果这两个特性都没有，则MSBuild先执行它遇到的第一个Target。Target有一个DependsOnTargets特性表示当前目标依赖另一个目标，效果就是DependsOnTargets特性指定的目标先于当前目标执行。

1 <?xml version="1.0" encoding="utf-8"?>

2 <!--目标执行顺序-->

3 <!--如果InitialTargets特性存在，则首先执行此目标列表-->

4 <!--如果DefaultTargets特性存在，则继续执行此目标列表-->

5 <Project InitialTargets="B1;B2" DefaultTargets="B3;B4"

6 ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">

7 <!--如果发现Target具有DependsOnTargets特性-->

8 <!--则先执行DependsOnTargets指定的目标-->

9 <!--MSBuild4新加入了RunBeforeTargets和RunAfterTargets特性-->

10 <!--其作用和DependsOnTargets类似，一前一后，不做演示了-->

11 <Target Name="B1" DependsOnTargets="B5">

12 <Message Text="B1"/>

13 </Target>

14 <Target Name="B2">

15 <Message Text="B2"/>

16 </Target>

17 <Target Name="B3">

18 <Message Text="B3"/>

19 </Target>

20 <Target Name="B4">

21 <Message Text="B4"/>

22 </Target>

23 <Target Name="B5">

24 <Message Text="B5"/>

25 </Target>

26 <!--结果为：B5 B1 B2 B3 B4-->

27 </Project>

**导入**

就是导入其他的构建文件，这样我们就可以不需要在一个文件中编写所有的构建配置了。

1. <Import Project="foo.proj" />

即可把引入的构建文件里的内容直接插入到所在的位置。

**条件**

可以在任何地方使用Condition来控制是否计算或执行一个属性、项、任务、目标和引入，也就是说你可以在任何你想要的地方通过Condition来进行条件的控制。

1. <PropertyGroup>
2. <Name>Alice</Name>
3. <IsDefaultName Condition=" '$(Name)' == 'alice' ">true</IsDefaultName>
4. <IsDefaultName Condition=" '$(Name)' != 'alice' ">false</IsDefaultName>
5. </PropertyGroup>

上述代码中，如果构建的时候Name是Alice，那么IsDefaultName就是true，否则是false。

**任务错误处理**

任务可能会发生错误，在MSBuild中，可以通过Error产生错误、Warn产生警告；一些内置的任务（例如Delete、Copy等）也可能产生错误；对于自行编写的任务而言，也有其方式产生错误或者警告。

<Target Name="Print">

<Message Text="你好，$(Name)，是否默认名字：$(IsDefaultName)" Importance="high" />

<Error Condition=" '$(IsDefaultName)' == 'false' " ContinueOnError="ErrorAndContinue" Text="发生错误了" />

</Target>

<Target Name="Build" DependsOnTargets="Print">

<Message Text="构建完了" Importance="high" />

</Target>

**结构化日志**

有时编写好了构建文件之后，我们希望能够查看整个构建流程或者失败的原因等，这个时候文本的日志就不够用了。在MSBuild中，我们有强大的结构化日志。

1. msbuild build.proj -bl:output.binlog

只需要构建的时候传入一个-bl参数指定binlog的位置，MSBuild就能在构建时为我们生成一个极其强大的结构化日志

**MSBuild特殊字符**

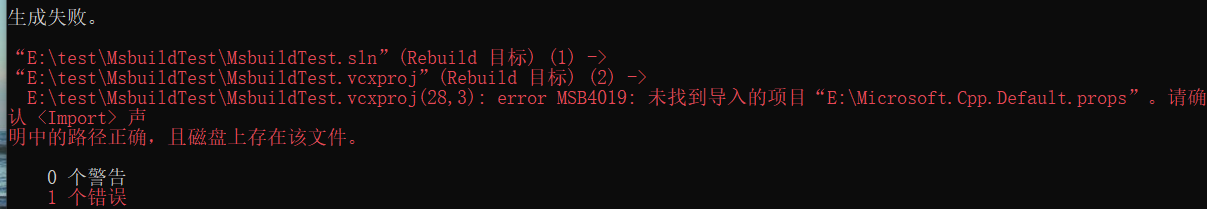
一些字符在MSBuild中代表着特殊的上下文含义，如下：

MSBuild的特殊字符[%引用元数据]、[$引用属性]、[@引用项]、['条件或其他表达式]、[;列表分隔符]、[?文件名通配符]、[\*文件名通配符]；

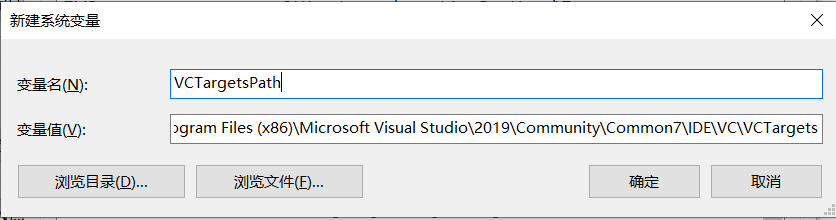
XML的保留字符：<、>、&、"、'

**使用MSBuild编译sln**

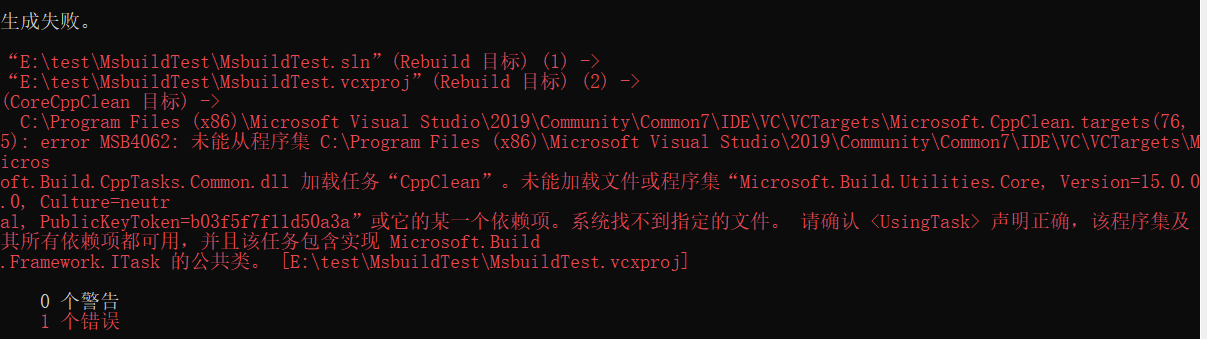
输入命令行：MSBuild MsbuildTest.sln /t:Rebuild /p:Configuration=Release

产生如下错误：  


查看.vcxproj猜测是无法解析VCTargetsPath的目录，百度查找将VCTargetsPath设置为环境变量



产生如下错误：



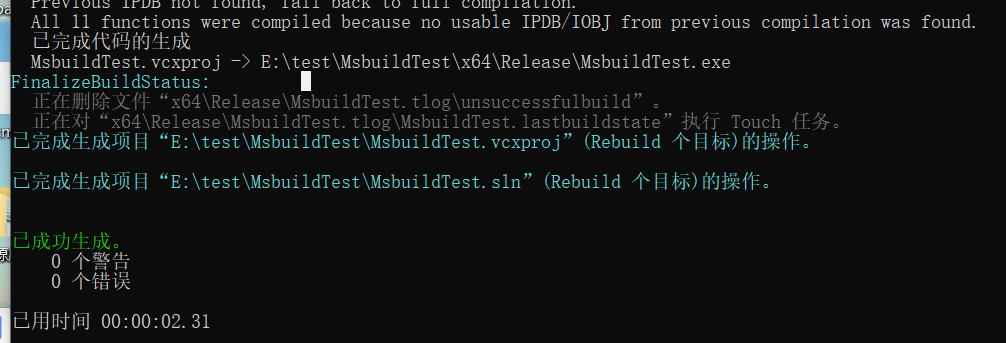
<https://qa.1r1g.com/sf/ask/1126451861/> 解决方案很多，都没有解决

查找百度MSBuild文件中缺少了Microsoft.Cpp文件

VCTargetsPath="C:\Program Files (x86)\MSBuild\Microsoft.Cpp\v4.0\V140"，目前没有办法解决缺少的Microsoft.Cpp文件，后续又复制了Microsoft.Cpp文件，依旧无法找到Microsoft.cpp.Default.props。添加了VCTargetsPath环境变量以及相应的bin环境变量都无法解决问题，也会导致vs无法编译，需要删除相关环境变量重启电脑才能恢复vs的编译。

以上解决方案都有问题，只需要在环境变量中添加C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\MSBuild\Current\Bin

最终编译成功



**MSBuild编译时使用的参数**

<http://www.javashuo.com/article/p-xccoirdp-dw.html> 相关参考

若是在文件夹里面存在多个不一样的sln文件等，在msbuild后面添加对应的文件

msbuild xx.sln

有多个项目一块儿编译，经过并行编译能够加快速度，用 /m 参数开启并行编译

msbuild /m

经过后面带上数字表示多少CPU同时编译，下面代码表示6个CPU同时编译

msbuild /m:6

经过发布版编译就是设置configuration属性为release经过下面代码

msbuild /p:configuration="release"

在msbuild经过 /p 设置对应的属性的值，在msbuild的参数是不区分 / 和 - 也就是 /p 和 -p 是相同

经过 -t:rebuild 从新编译

msbuild -t:rebuild

经过 -fileLogger 或 -fl 能够指定输出到文件，经过 -filelogparameters 或短参数 flp 能够指定输出的日志文件

msbuild -fl -flp:logfile=xx.log;verbosity=n

这里的verbosity表示输出等级，运行上面代码建议编译当前文件夹里面的项目，而后将编译日志输出到xx.log文件夹

经过 -t:restore 能够还原包

msbuild -t:restore

经过 -t:clean 清理项目

msbuild -t:clean

经过 /t:pack 打包

msbuild -t:pack