

TEAMPRISE EXTENSIONS FOR TEAM FOUNDATION BUILD

OVERVIEW

The following instructions are designed to help you build a Java application from a Microsoft Visual Studio Team Foundation Server Build (Team Build). Team Build is only able to execute MSBuild scripts to perform builds. Team Build creates a master build file - a bit like a boot strap build file. The default Team Build scripts have some useful functionality for labeling, downloading source, updating work items, calculating changes etc. For .NET based builds, the default script then fires off a child MSBuild process to build the .NET projects. The Teamprise Extensions for Team Foundation Build provide a custom MSBuild target allows you to easily call Ant from with-in the TFSBuild.proj file created by Team Build. As well as making the calling of Ant easier, the task reports data of the build progress into TFS. The Team Build portions of Team Foundation Server has had major overhaul as part of TFS 2008, therefore there are two versions of the custom target, one for TFS 2005 and for TFS 2008.

PREREQUISITES

You must have the following installed on the same server as your TFS Build Agent:-

- Java JDK (the latest one from Sun is recommended)
- Ant (the latest version is recommended, Ant 1.7.0 had some changes that make the running of JUnit tests easier).

INSTALLING THE TEAMPRISE EXTENSIONS FOR TEAM FOUNDATION BUILD.

The Teamprise Extensions for Team Foundation Build consist of the follow components:-

- Teamprise.Build.Ant.targets file, that in turn uses;
- Teamprise Build tasks assembly (Teamprise.Build.dll)

Due to the differences in the build functionality provided by TFS 2005 and TFS 2008, there are current two versions of these, v1 and v2 for TFS2005 and TFS 2008 respectively.

USING THE INSTALLER

By far the easiest way to install the build extensions is to run the TeampriseBuildExtensions installer (msi) on the machine running the Team Foundation Build Server process (the build agent). This will install the necessary components into the MSBuild extensions path – usually located at %ProgramFiles%\MSBuild.

MANUAL INSTALLATION

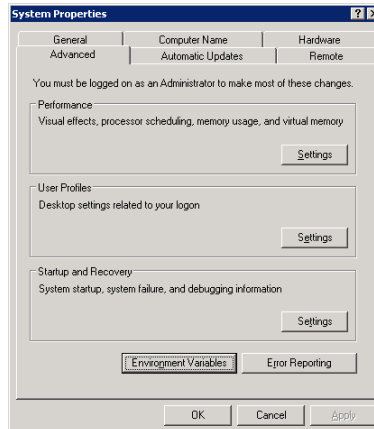
A manual installation zip archive is provided. If using this method, the contents of the archive should be extracted to disk and then the contents of the targets folder copied to %ProgramFiles%\MSBuild\Teamprise.

POST INSTALLATION TASKS

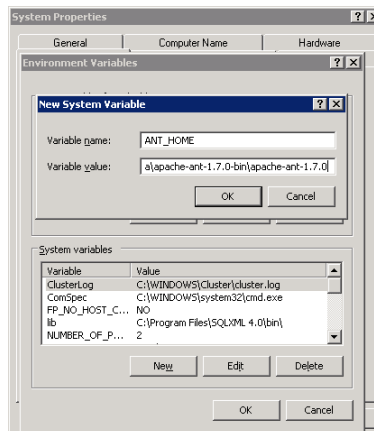
Once you have installed the Teamprise Extensions for Team Foundation Build onto the build agent, you will need to configure the machine to provide the locations of Ant and your desired JDK. This is performed by setting the following environment variables at the machine level.

- ANT_HOME, The location of your chosen version of Ant.
- JAVA_HOME, The version of Java that you wish to use to perform your builds. Note that you do not need to configure JAVA_HOME if the default JDK for your system is already installed and it is the JDK that you wish to compile your builds with.

To set environment variables, go to Control Panel, System, Advanced and press the Environment Variables button.



Then, in the System Variables section, press New and enter the environment variable value.



ANT TASK REFERENCE

The Ant task works by calling Java to run Ant. The task first parses the Ant file to locate the name of the project and the description. It then calls Ant and the resulting output from Ant is then parsed by the task to look for key information (such as javac and junit tasks) as well as to pass the results into the MSBuild log. Options which are normally available via ant launching script are available as additional attributes to the Ant task.

EXAMPLE USAGE

```
<Ant TeamFoundationServerUrl="$(TeamFoundationServerUrl)"
    BuildFile="$(SolutionRoot)\java\HelloWorld\build.xml"
    BuildUri="$(BuildUri)"
    AntHome="C:\Java\apache-ant-1.7.0-bin\apache-ant-1.7.0"
    JavaHome="C:\Java\jdk1.6.0_02"
    Flavor="%ConfigurationToBuild.FlavorToBuild"
    Platform="%ConfigurationToBuild.PlatformToBuild"
    Properties="BinariesRoot=$(BinariesRoot);BuildDefinitionName=$(BuildDefinitionName)
;BuildDefinitionUri=$(BuildDefinitionUri);BuildDirectory=$(BuildDirectory);BuildNum
ber=$(BuildNumber);DropLocation=$(DropLocation);LogLocation=$(LogLocation);SourceGe
tVersion=$(SourceGetVersion);TestResultsRoot=$(TestResultsRoot);TeamProject=$(TeamP
roject);workspaceName=$(workspaceName);workspaceOwner=$(workspaceOwner)"
/>
```

TASK REFERENCE

The following is a complete list of all the attributes supported by the task, note that many of them are best left to the default values unless a different behavior is explicitly required. Items that are in bold are the ones that are frequently used.

Parameter	Required	Description
AntHome	No	Location of Ant on Build Server. If not specified then the value of the ANT_HOME environment variable will be used.
AutoProxy	No	In Java 1.5+, use the OS proxies
BuildFile	No	Name of the build file to use, by default this is "build.xml" in the current directory.
BuildUri	Yes	The team system URI which uniquely represents the instance of the build being run. With-in a Team Build MSBuild script, this is normally available in the MSBuild property \$(BuildUri)
Debug	No	Set to "true" to instruct Ant to print debugging information. By default this is set to "false".
Flavor	No	The flavor of the build i.e. Release, Debug etc. This will default to "Release". In the Team Build MSBuild scripts, this is normally available as the global property %(ConfigurationToBuild.FlavorToBuild)
InputHandler	No	Specifies the Ant class which will handle input requests
JavaHome	No	Location of Java home directory on build server. If not specified then the value of the JAVA_HOME environment variable will be used.
KeepGoing	No	Instruct Ant to execute all targets that do not depend on failed target(s)
Lib	No	Specifies a path for Ant to search for jars and classes.
Listener	No	Add an instance of an Ant class as a project listener
Logger	No	Specify an Ant class to perform logging.

Parameter	Required	Description
Main	No	Override Ant's normal entry point with specified Ant class.
NoClasspath	No	Run ant without using CLASSPATH
Noinput	No	Do not allow interactive input in Ant script
NoJavacBuildSteps	No	Set to "true" to suppress the reporting of javac steps to TFS. By default javac steps are added as build steps.
NoUserLib	No	Run ant without using the jar files from \${user.home}/.ant/lib
Platform	No	The build platform i.e. Any CPU, x86, x64. This will default to "Any CPU". In the Team Build MSBuild script, this is normally available as the global property %(ConfigurationToBuild.PlatformToBuild)
Properties	No	Properties to pass to Ant in "name=value;name2=value2" syntax. When calling Ant, it is often useful to pass through properties from the originating MSBuild script – for example Properties="BinariesRoot=\$(BinariesRoot);BuildDefinitionName=\$(BuildDefinitionName);BuildDefinitionUri=\$(BuildDefinitionUri);BuildDirectory=\$(BuildDirectory);BuildNumber=\$(BuildNumber);DropLocation=\$(DropLocation);LogLocation=\$(LogLocation);SourceGetVersion=\$(SourceGetVersion);TestResultsRoot=\$(TestResultsRoot);TeamProject=\$(TeamProject);WorkspaceName=\$(WorkspaceName);WorkspaceOwner=\$(WorkspaceOwner)"
PropertyFile	No	Instruct Ant to load all properties from file with -D properties taking precedence
Target	No	Single Ant Target to execute. If not specified then the default target specified in the script will be used. It is often useful to specify a target that is executed by Team Build and leave the default target to be what would get executed by a developer in a local workstation build.
Targets	No	Comma separated list of ant targets to execute.
TeamFoundationServerUrl	Yes	The URL of the Team Foundation Server to talk to. In the Team Build MSBuild script, this is often available in the property \$(TeamFoundationServerUrl)
Verbose	No	Set to "true" to instruct Ant to be extra verbose.