



Software Builds from Harvest using Apache's Ant

1 Document Version

Revision	Editor	Reviewer	Comments
1.0	Tom Clark		Initial draft.



2 Table of Contents

1	Document Version	2
2	Table of Contents.....	3
3	Ant overview	4
4	Integration with Harvest	4

3 Ant overview

Ant is a popular build tool created as part of the Apache project. It was originally written specifically to build the Java-servlet container Tomcat, which is also part of the Apache project. Ant is now widely used to create builds of various types of software written in a variety of different programming languages. Its task-based, modular architecture allows it to be easily extended for different requirements.

Ant uses an XML configuration file to define a build process. The main components of an Ant configuration script are 'properties', 'targets' and 'tasks'. A property is a variable used in the script, a task is an individual action in the script (for example, creating a directory or compiling some code), and a target is a collection of tasks. Targets can be nested to achieve more complex builds and dependencies.

4 Integration with Harvest

Ant contains a large number of custom tasks, the functions of which range from compiling specific types of code to extracting code from specific code repositories. However, there is no custom task defined in Ant for extracting code from Harvest. This problem can easily be resolved by using Ant's <exec> task to run Harvest's command-line check-out utility, hco.

Consider the following fairly typical build process:

- Prepare for build
- Extract code
- Compile code
- Package build for deployment
- Tidy up

Using Harvest as the repository for the codebase, this simple build might be achieved with the following Ant script:

```
1 <project name="MyProject" default="build" basedir=". ">
2   <description>
3     simple build file
4   </description>
5
6   <!-- set global properties for the build -->
7   <property name="src.dir" value="c:\src"/>
8   <property name="build.dir" value="c:\build"/>
9   <property name="archive.dir" value="c:\archive"/>
10
11  <property name="win.os" value="Windows 2000"/>
12
13  <!-- Properties for use with hco -->
14  <property name="har.server" value="harserver"/>
15  <property name="har.user" value="deployer"/>
16  <property name="har.password" value="deployer"/>
17  <property name="har.environment" value="environment"/>
18  <property name="har.pc" value="pc"/>
19  <property name="har.state" value="'Unit Testing'"/>
20  <property name="har.filepattern" value="*"/>
21  <property name="har.log" value="E:\logs"/>
22
23  <target name="build"
24    depends="init,extract,compile,archive,clean"/>
25
26  <target name="init">
27    <tstamp/>
28    <mkdir dir="${src.dir}"/>
29    <mkdir dir="${build.dir}"/>
30  </target>
31
32  <target name="extract" depends="init">
33
34    <exec executable="D:\harvest\CA\CCC_Harvest\hco.exe"
35      os="${win.os}" dir="${src.dir}">
36      <arg line="-b ${har.server}"/>
37      <arg line="-usr ${har.user}"/>
38      <arg line="-pw ${har.password}"/>
39      <arg line="-op ${har.pc}"/>
40      <arg line="-en ${har.environment}"/>
41      <arg line="-st ${har.state}"/>
42      <arg line="-vp \Viewpath"/>
43      <arg line="-br"/>
44      <arg line="-r"/>
45      <arg line="${har.filepattern}"/>
46    </exec>
47
48  </target>
49
50  <target name="compile" depends="init">
51    <csc targetType="exe" incremental="false"
52      destFile="MyApp.exe">
```

```
53     <src dir="${src.dir}" includes="*.cs"/>
54     </csc>
55 </target>
56
57 <target name="archive" depends="compile">
58     <mkdir dir="${archive.dir}"/>
59     <zip destfile="${archive.dir}/MyProject-${DSTAMP}.zip"
60         basedir="${build.dir}"/>
61 </target>
62
63 <target name="clean" depends="archive">
64     <delete dir="${build.dir}"/>
65     <delete dir="${src.dir}"/>
66 </target>
67 </project>
```

The relevant parts of this script are the properties used as parameters to hco during the exec task, and the exec task itself. The properties are set up on lines 11 – 21 and the exec task is on lines 34 – 46. Note that some lines have been wrapped for clarity.

These instructions may also be compatible with NAnt which is a .NET clone of Ant.

Hopefully, this should give the reader a clearer understanding of how to use Harvest's hco.exe from Ant. Further integration with Harvest could be achieved through the use of Trinem's middleware component, Trilogy, for example to run a build process entirely from a client Workbench, passing the build environment, state and user to Ant from a Harvest User Defined Process (UDP).

For more information about Trilogy, see the Trinem website at www.trinem.com.