The Apache Ant Project

A Dive Into A Powerful Scripting Tool

Apache Ant

- Ant is a java library and command line tools focused on building and deploying files.
- It is extensible and has grown to far more than a build tool.
- It is an active project that was originally released in 2000.

DEVELPRENEUR
BUILDING BETTER DEVELOPERS

Our Focus

- Installation/Getting Started
- General format and properties
- Copying files
- Tar/Zip files
- Variables
- Secure copy
- Executing SQL Scripts
- Using Git
- Firing off Javadoc
- Executing Tests

DEVELPRENEUR
BUILDING BETTER DEVELOPERS

Installation

- Download the latest version at: https://ant.apache.org/bindownload.cgi
- It is a Java application so you need a minimum JVM as well. Looks for that in the requirements. Most current ones need at least Java 5.
- Note where you install and configure the environment variable ANT_HOME to make things easier.

General Format

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="Develpreneur" default="all" basedir=".">
  <target name="Basic" description="This is just a very simple task to display
a message">
    <echo message="About to run the most basic script."/>
    <tstamp/>
    <echo message="Finished build at ${TSTAMP}"/>
  </target>
  <target name="clean" description="Remove any non xml files just to be safe">
    <echo message="Removing extra files (not *.xml) ..."/>
    <delete>
      <fileset dir=".">
        <include name="*.backup"/>
      </fileset>
    </delete>
  </target>
  <target name="all" description="run everything" depends="Basic,clean">
    <echo message="Finished build at ${TSTAMP}"/>
  </target>
</project>
```

An XML File

- Project: Name, Default, Basedir
- Echo sends a message to the screen
- Target descriptions to help plus main vs other tasks
- Depends to link tasks together (runs left to right)

Copying Files

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project name="Develpreneur" default="all" basedir=".">
  <target name="Basic" description="This is just a very simple task to display a message">
    <echo message="About to run the most basic script."/>
    <tstamp/>
    <echo message="Finished build at ${TSTAMP}"/>
  </target>
  <target name="clean" description="Remove any non xml files just to be safe">
    <echo message="Removing extra files (not *.xml) ..."/>
    <delete>
      <fileset dir=".">
        <include name="*.backup"/>
      </fileset>
    </delete>
  </target>
  <target name="all" description="run everything" depends="Basic,clean">
    <echo message="Finished build at ${TSTAMP}"/>
  </target>
  <target name="backup" description="copy the basic.xml file to backup" depends="clean">
    <copy file="basic.xml" tofile="basic.backup"/>
  </target>
  <target name="backupchange" description="copy the basic.xml file to backup if it is newer">
    <copy file="basic.xml" tofile="basic.backup"/>
  </target>
  <target name="copynewdir" description="Copy all xml files to a new folder">
    <copy failonerror="false" todir="newfolder">
      <fileset dir=".">
        <include name="**/*.xml"/>
        <exclude name=".backup"/>
      </fileset>
    </copy>
  </target>
</project>
```

Group/Compress

```
  <target name="pkg" description="Package files into tarball"
depends="Basic,clean">
    <echo message="Packaging files..."/>
    <delete file="./pkg.tar"/>
    <tar destfile="./pkg.tar" basedir="."/>
  </target>
  <target name="zip" description="Package files into zipfile"
depends="Basic,clean">
    <echo message="Zipping files..."/>
    <delete file="./pkg.zip"/>
    <zip destfile="./pkg.zip" basedir="."/>
  </target>
```

Script Variables

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="Develpreneur" default="all" basedir=".">
  <property name="src.dir" value="."/>
  <property name="target.dir" value="newfolder"/>
  <property name="server" value="rob@rbprod:/home/rob"/>
  <target name="Basic" description="This is just a very simple task to
display a message">
    <echo message="About to run the most basic script."/>
    <tstamp/>
    <echo message="Finished build at ${TSTAMP}"/>
  </target>
  <target name="clean" description="Remove any non xml files just to be
safe">
    <echo message="Removing extra files (not *.xml) ..."/>
    <delete>
      <fileset dir="${src.dir}">
        <include name="*.backup"/>
      </fileset>
    </delete>
  </target>
```

DEVELPRENEUR
BUILDING BETTER DEVELOPERS

Ant Libraries

- SCP requires the ant-jsch.jar file.
- One location is http://www.java2s.com/Code/Jar/a/Downloadantjsch181jar.htm
- Copy the jar file into the ant/lib folder
- This is where you put database drivers as well

An SCP Example

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="TimeDemo">
  <!-- Loading application directory properties -->
  <property name="deploy.dir" value="/Library/WebServer/Documents/TimeDemo"/>
  <property name="src.dir" value="/Users/robbroadhead/Coding/TimeDemo/WebMain"/>
  <property name="root.dir" value="/Users/robbroadhead/Coding/TimeDemo"/>
  <property name="repo" value="https://rbroadhead@bitbucket.org/rbroadhead/timematters.git"/>
  <property name="server" value="rob@mytime:/home/rob"/>
  <target name="all" depends="clean,deploy">
    <echo message="Ran full build."/>
    <tstamp/>
    <echo message="Finished build at ${TSTAMP}"/>
  </target>
  <target name="deploy" description="Copy the changed files to the target">
    <copy failonerror="false" todir="${deploy.dir}">
      <fileset dir="${src.dir}">
        <include name="**/*"/>
        <exclude name=".*"/>
      </fileset>
    </copy>
  </target>
  <target name="pkg" description="Package files into tarball" depends="deploy">
    <echo message="Packaging files..."/>
    <delete file="${root.dir}/timematters.tar"/>
    <tar destfile="${root.dir}/timematters.tar" basedir="${deploy.dir}"/>
    <echo message="Sending files..."/>
    <scp file="${root.dir}/timematters.tar" todir="${server}" keyfile="/Users/robbroadhead/.ssh/id_rsa"
trust="true"/>
  </target>
  <target name="clean" description="Remove the files just to be safe">
    <echo message="Removing old files..."/>
    <delete>
      <fileset dir="${deploy.dir}">
        <exclude name="images/*"/>
      </fileset>
    </delete>
  </target>
```

Description

Copies a file or FileSet to or from a (remote) machine running an SSH daemon. FileSet *only* works for copying files from the local machine to a remote machine.

Note: This task depends on external libraries not included in the Ant distribution. See Library Dependencies for more information. This task has been tested with `jsch-0.1.2` and later.

See also the sshexec task

Parameters

Attribute	Description	Required
file	The file to copy. This can be a local path or a remote path of the form `user[:password]@host:/directory/path`. `:password` can be omitted if you use key based authentication or specify the *password* attribute. The way remote path is recognized is whether it contains "@" character or not. This will not work if your *localPath* contains "@" character.	Yes, unless a nested `<fileset>` element is used
localFile	This is an alternative to the *file* attribute. But this must always point to a local file. The reason this was added was that when you give file attribute it is treated as remote if it contains "@" character. This character can exist also in local paths. *since Ant 1.6.2*	Alternative to *file* attribute
remoteFile	This is an alternative to the *file* attribute. But this must always point to a remote file. *since Ant 1.6.2*	Alternative to *file* attribute
todir	The directory to copy to. This can be a local path or a remote path of the form `user[:password]@host:/directory/path`. `:password` can be omitted if you use key based authentication or specify the *password* attribute. The way remote path is recognized is whether it contains "@" character or not. This will not work if your *localPath* contains "@" character.	Yes
localTodir	This is an alternative to the *todir* attribute. But this must always point to a local directory. The reason this was added was that when you give *todir* attribute it is treated as remote if it contains "@" character. This character can exist also in local paths. *since Ant 1.6.2*	Alternative to *todir* attribute
localToFile	Changes the file name to the given name while receiving it, only useful if receiving a single file. *since Ant 1.6.2*	Alternative to *todir* attribute
remoteTodir	This is an alternative to the *todir* attribute. But this must always point to a remote directory. *since Ant 1.6.2*	Alternative to *todir* attribute
remoteToFile	Changes the file name to the given name while sending it, only useful if sending a single file. *since Ant 1.6.2*	Alternative to *todir* attribute
port	The port to connect to on the remote host.	No; defaults to "22"
trust	This trusts all unknown hosts if set to "yes" or "true". **Note:** If you set this to "false" (the default), the host you connect to must be listed in your *knownhosts* file, this also implies that the file exists.	No; defaults to "no"
knownhosts	This sets the known hosts file to use to validate the identity of the remote host. This must be a SSH2 format file. SSH1 format is not supported.	No; defaults to `${user.home}/.ssh/known_hosts`
failonerror	Whether to halt the build if the transfer fails.	No; defaults to "true"
password	The password.	Yes, unless you are using key based authentication or the password has been given in the file or *todir* attribute
keyfile	Location of the file holding the private key.	Yes, if you are using key based authentication
passphrase	Passphrase for your private key.	No; defaults to an empty string
verbose	Determines whether SCP outputs verbosely to the user. Currently this means outputting dots/stars showing the progress of a file transfer. *since Ant 1.6.2*	No; defaults to "false"
sftp	Determines whether SCP uses the sftp protocol. The sftp protocol is the file transfer protocol of SSH2. It is recommended that this be set to "true" if you are copying to/from a server that doesn't support scp1. *since Ant 1.7*	No; defaults to "false"
preserveLastModified	Determines whether the last modification timestamp of downloaded files is preserved. It only works when transferring from a remote to a local system and probably doesn't work with a server that doesn't support SSH2. *since Ant 1.8.0*	No; defaults to "false"
filemode	A 3 digit octal string, specify the user, group and other modes in the standard Unix fashion. Only applies to uploaded files. Note the actual permissions of the remote file will be governed by this setting and the UMASK on the remote server. *since Ant 1.9.5.*	No; default is "644"
dirmode	A 3 digit octal string, specify the user, group and other modes in the standard Unix fashion. Only applies to uploaded dirs. Note the actual permissions of the remote dir will be governed by this setting and the UMASK on the remote server. *since Ant 1.9.5.*	No; default is "755"
serverAliveIntervalSeconds	Sets a timeout interval in seconds after which if no data has been received from the server, the task will send a message through the encrypted channel to request a response from the server. *since Ant 1.9.7*	No, the default is "0", indicating that these messages will not be sent to the server
serverAliveCountMax	The number of server alive messages which may be sent without receiving any messages back from the server. Only used if *serverAliveIntervalSeconds* is not "0". *since Ant 1.9.7*	No; defaults to "3"
compressed	Whether to enable compression during transfer. *since Ant 1.9.8*	No; defaults to "false"

"When in doubt, do a search on '[task name] ant task' to find the documentation and examples quickly. Most commands are very similar in properties and options"

DEVELPRENEUR
BUILDING BETTER DEVELOPERS

A SQL Example

```
  <target name="dbreset" description="Drop and rebuild the database
from the script">
    <echo message="Dropping and rebuilding the database"/>
    <sql driver="com.mysql.jdbc.Driver" url="jdbc:mysql://127.0.0.1/
mydb" userid="dbuser" password="password123" src="${src.dir}/sql/
MyDBSource.sql"/>
    <sql driver="com.mysql.jdbc.Driver" url="jdbc:mysql://127.0.0.1/
mydb" userid="dbuser" password="pasword123"><![CDATA[
update some_table set column1 = column1 + 1 where column2 < 42;
]]></sql>
  </target>
```

DEVELPRENEUR

BUILDING BETTER DEVELOPERS

A Git Example

```
<!-- Git Marcos-->
<macrodef name="git">
  <attribute name="command"/>
  <attribute name="dir" default=""/>
  <element name="args" optional="true"/>
  <sequential>
    <echo message="git @{command}"/>
    <exec executable="git" dir="@{dir}">
      <arg value="@{command}"/>
      <args/>
    </exec>
  </sequential>
</macrodef>
<macrodef name="git-clone-pull">
  <attribute name="repository"/>
  <attribute name="dest"/>
  <sequential>
    <git command="clone">
      <args>
        <arg value="@{repository}"/>
        <arg value="@{dest}"/>
      </args>
    </git>
    <git command="pull" dir="@{dest}"/>
  </sequential>
</macrodef>
<target name="grabsource" description="Pull files from bitbucket">
  <echo message="Getting files from repository ${repo}"/>
  <git-clone-pull repository="${repo}" dest="source"/>
  <echo message="Finished pull at ${TSTAMP}"/>
</target>
```

DEVELPRENEUR

BUILDING BETTER DEVELOPERS

Javadoc Example

```
 <target name="doc" description="generate documentation">
    <javadoc packagenames="com.*" sourcepath="${java.dir}"
destdir="${doc.dir}" author="true" version="true" use="true"
windowtitle="Test API">
      <doctitle><![CDATA[<h1>Ant Tutorial Docs</h1>]]></
doctitle>
      <bottom><![CDATA[<i>Copyright &#169; 2018 My Corp. All
Rights Reserved.</i>]]></bottom>
      <tag name="todo" scope="all" description="To do:"/>
      <link href="https://docs.oracle.com/javase/8/docs/api/"/
>
    </javadoc>
  </target>
```

JUnit Example

```
<junit printsummary="yes" fork="yes" haltonfailure="yes">
    <formatter type="plain"/>
    <test name="my.app.TestCase1"/>
    <test name="my.app.TestCase2"/>
</junit>
```

Masking Your Scripts

- Take it step by step, add a task here or there. No need to go crazy.
- Get comfortable with some core tasks.
- Expand as needed.
- The documentation and examples are your friends.

DEVELPRENEUR
BUILDING BETTER DEVELOPERS

Excellent Uses

- Backup scripts
- Database updates, imports and exports
- Automating common admin tasks
- Easier than shell scripts and can be as powerful
- Cross platform so perfect when developing and deploying on different ones

Thank You!

I appreciate your time and would love to discuss any of this further. You can send questions, comments and suggestions through any of these methods.

- info@develpreneur.com
- https://develpreneur.com/contact-us
- @develpreneur
- https://www.facebook.com/Develpreneur

Our goal is building better developers from where they are today for a better tomorrow.

DEVELPRENEUR
BUILDING BETTER DEVELOPERS