Apache Ant Java build tool

START HERE

`ant`
Run default target

`ant -Dname=value`
Set a property

`ant <target>`
Run one target

`ant clean jar`
Chain targets

`ant -p`
List documented targets

`ant -v`
Verbose output

`ant -f <file>`
Use another build file

`ant -version`
Show Ant version

CONCEPTS

Ant reads build.xml in the current directory. Run with no target and it runs the project's default target.

Project build.xml, default target
- **Target** named step, depends on others
  - **Task** javac, jar, copy ...

Properties are immutable: the first definition wins, so -D on the command line overrides build.xml.

depends only orders targets. It does not rerun a target that already ran in this invocation.

RUN

`ant`
Run the project's default target

`ant <target> [<target>...]`
Run targets in order

`ant -projecthelp / ant -p`
List targets that have a description

`ant -buildfile <file> / ant -f <file>`
Use a different build file

`ant -find [<file>] / ant -s [<file>]`
Search parent dirs for the build file

`ant -version`
Print Ant version

`ant -diagnostics`
Print Java, classpath and Ant diagnostics

`ant -help / ant -h`
Show command line options

`ant -keep-going / ant -k`
Continue with targets not depending on a failure

`ant -noinput`
Never prompt for input (CI)

PROPERTIES AND OUTPUT

`ant -D<name>=<value>`
Set a property from the command line

`ant -propertyfile <file>`
Load properties from a file
-D values still take precedence over the file.

`ant -quiet / ant -q`
Less output

`ant -silent / ant -S`
Only task output and failures

PROPERTIES AND OUTPUT (cont.)

`ant -verbose / ant -v`
Extra output

`ant -debug / ant -d`
Debugging output

`ant -emacs / ant -e`
Log without [task] adornments

`ant -logfile <file> / ant -l <file>`
Write the log to a file

`ant -logger <class>`
Use a custom logger class

`ant -listener <class>`
Add a project listener

`ant -lib <path>`
Add jars or dirs to Ant's classpath

`ant -nouserlib`
Skip jars in ~/.ant/lib

`ant -noclasspath`
Ignore the CLASSPATH variable

`ant -autoproxy`
Use OS proxy settings

`ant -nice <1-10>`
Set main thread priority

`ANT_OPTS="-Xmx1g" ant`
Pass options to the JVM

`ANT_ARGS="-noinput"`
Default Ant arguments from environment

BUILD FILE BASICS

`<project name="<n>" default="<target>" basedir=".">`
Root element

`<target name="<n>" depends="<a>,<b>" description="<text>">`
Define a target
Only targets with a description show in ant -p.

`<property name="<n>" value="<v>"/>`
Define a property

`<property file="build.properties"/>`
Load properties from a file

`<property environment="env"/>`
Read env vars as ${env.NAME}

`${<name>}`
Reference a property

`${basedir}`
Project base directory

`<target name="<n>" if="<prop>">`
Run only if a property is set

BUILD FILE BASICS (cont.)

`<target name="<n>" unless="<prop>">`
Run only if a property is not set

`<import file="<other.xml>"/>`
Include another build file

`<macrodef name="<n>">`
Define a reusable task

`<path id="<id>"><fileset dir="lib" includes="*.jar"/></path>`
Reusable classpath

`<fileset dir="<d>" includes="**/*.java" excludes="<pat>"/>`
Select files by pattern

`<echo message="<text>"/>`
Print a message

`<exec executable="<cmd>"><arg value="<a>"/></exec>`
Run an external command

`<antcall target="<n>"/>`
Call a target (new context)

COMMON TASKS

`<mkdir dir="<d>"/>`
Create a directory

`<delete dir="<d>"/>`
Remove a directory tree

`<copy todir="<d>"><fileset dir="<s>"/></copy>`
Copy files

`<move file="<f>" tofile="<t>"/>`
Move or rename a file

`<javac srcdir="src" destdir="build/classes" classpathref="<id>" includeantruntime="false"/>`
Compile Java sources
Set includeantruntime="false" to silence the runtime warning.

`<jar destfile="dist/<n>.jar" basedir="build/classes"/>`
Build a jar

`<manifest><attribute name="Main-Class" value="<cls>"/></manifest>`
Set jar manifest (inside <jar>)

`<java classname="<cls>" classpathref="<id>" fork="true"/>`
Run a Java class

`<junit printsummary="yes" haltonfailure="yes">`
Run JUnit tests
Needs the junit task and ant-junit jar; JUnit 5 uses <junitlauncher>.

COMMON TASKS (cont.)

`<junitlauncher>`
Run JUnit Platform tests (Ant 1.10.6+)

`<javadoc sourcepath="src" destdir="docs"/>`
Generate API docs

`<zip destfile="<f>.zip" basedir="<d>"/>`
Create a zip archive

`<war destfile="<f>.war" webxml="<web.xml>"/>`
Build a war

`<get src="<url>" dest="<file>"/>`
Download a file

`<replace file="<f>" token="<a>" value="<b>"/>`
Text replace in a file

`<tstamp/>`
Set DSTAMP, TSTAMP, TODAY

`<fail message="<msg>"/>`
Fail the build

RECIPES

Minimal build.xml

```
<project name="app"
default="jar" basedir=".">
  <target name="compile">
    <mkdir dir="build/classes"/
>
    <javac srcdir="src"
destdir="build/classes"
includeantruntime="false"/>
  </target>
  <target name="jar"
depends="compile"
description="Package">
    <jar destfile="dist/
app.jar" basedir="build/
classes"/>
  </target>
  <target name="clean"
description="Remove output">
    <delete dir="build"/
><delete dir="dist"/>
  </target>
</project>
```

Clean build with a property

```
ant -p
ant clean jar -Dversion=1.2
# in build.xml: <jar
destfile="dist/app-
${version}.jar" .../>
```

Runnable jar and run it

```
<jar destfile="dist/app.jar"
basedir="build/classes">
  <manifest><attribute name=
"Main-Class" value="com.
example.Main"/></manifest>
</jar>
java -jar dist/app.jar
```

Debug a failing build

```
ant -v <target>
ant -d -l build.log <target>
ant -diagnostics
```