InstallTroubleshooting

Troubleshooting Installations

Here are core problems that cause grief, raise support calls, and how to fix them.

Testing your installation

After installing, a quick test of Ant makes sure that everything is well.

Test Ant is on the path

ant

> If this is gives some "unknown command" error, set your path up right.

Test ant version is correct

ant -version

The version printed must match what you have installed

{{{> ant -version Apache Ant version 1.7alpha compiled on May 24 2005}}}

If there is a different version printed, you have an old version somewhere.

Test that Ant is happy

ant -diagnostics

This triggers a deep analysis and health check of Ant and its environment. Things tested include

- a verification that the versions of the core and optional libraries match
- that all standard tasks can be instantiated; ones that can not, usually due to missing libraries, are listed
- what XML parser is used, and where from
- what libraries are in ANT_HOME/lib
- what your temp dir is, whether it is writeable, and how much a write to that dir is offset from the system time. Read only temp directories, or those on servers with different clocks, are not supported by Ant.
- what your classpath is
- what your system properties are

Common problems

CLASSPATH

Badly configured CLASSPATH environments cause no end of grief. Ant does not need a classpath, and nor should other modern Java programs. Make sure the environment variable is unset or empty. You can run Ant without the contents of CLASSPATH by going ant -noclasspath

CLASSPATH with quotes and spaces in

You should not have quotes inside your CLASSPATH variable. It is an error to do so. It is acceptable to have spaces at the beginning and end of the variable, if you have spaces in directory names. Better yet: Do not install stuff into directories with spaces.

JPackage RPM/Standalone Ant conflict

The JPackage team package up Ant and release it as an RPM, which is then redistributed in Linux distros such as RedHat and SuSE. If you want this integration/configuration management system, stick with it. Do not mix JPackage and Ant distributions. In particular, if you want to upgrade a JPackage installation, check the JPackage site for a new release, or remove their distribution and install Ant by hand.

Not enough JAR files for a task, AKA "Why doesn't <junit> work?"

Many of Ant's tasks depend on third party libraries. Ant does not include any of these, other than the Xerces XML parser. It is not enough to have the relevant optional JAR, such as "ant-junit.jar", you need the library that it depends upon, in this case "junit.jar". Similarly, it is not enough to have junit somewhere on your hard drive; ant is not going to look for it. JAR files must be

- in ANT_HOME/lib
- in ${user.home}/.ant/lib
- in a location specified by -lib on the command line

No third party JARs, no third party tasks.