Compiler Construction

Ant

Lennart Andersson

Revision 2011-01-21

2010

Build utilities

Specifies dependencies between targets and commands for building targets.

- **make** Feldman 1977. Shell specific.
- **ant** Davidson 2000. Java based and portable. XML syntax.

`ant` is not on the PATH by default. It is in `/usr/local/cs/bin`. Put the following line in `~/.zshrc`

```
alias ant=/usr/local/cs/bin/ant
```

or

```
PATH=/usr/local/cs/bin:$PATH
```


A Makefile

```
LECTURE = F01

.PHONY: clean

beamer.pdf:  slides.tex beamer.tex slides.sty
        pdflatex  beamer

$(LECTURE).pdf: beamer.pdf
        pdfnup --nup 2x2 --orient landscape --frame true \
        --scale 0.85 --outfile $(LECTURE).pdf beamer.pdf

clean:
        @rm -f *.dvi *.aux *.toc *.log  *.out *.nav *.snm
```


An ant build file

```
<project name="Compiler" default="compile">
  <target name="compile">
    <javac srcdir="src" destdir="bin">
    </javac>
  </target>
</project>
```

- A *project* may have a *name* and must have a *default target*.
- A target must have a *name* and should contain *tasks*.
- The compile target has one task that will compile all files in the src directory into the bin directory using javac.

XML syntax

A build file uses XML grammar. It contains properly nested *elements*. An element has a *start* and an *end tag*. A start tag has a *name* and zero or more *attributes*.

```
<project name="Compiler" default="compile">
```

An attribute has an attribute *name* and a string *value*.
An end tag has just the name preceded by a backslash.

```
</project>
```


XML syntax

If the element does not contain other elements then the start and end tags may be combined.

```
<javac srcdir="src" destdir="bin">
</javac>
```

is equivalent to

```
<javac srcdir="src" destdir="bin" />
```


A clean target

There are more than 80 predefined standard targets, see Ant tasks at `http://ant.apache.org/manual`.
The *delete* task below will delete all class files in the subdirectories of *bin*.

```
<target name="clean">
  <delete>
    <fileset dir="bin" includes="**/*.class"/>
  </delete>
</target>
```

The initial extra * will search the directory recursively.

XML syntax

A target may *depend* on a comma separated list of other targets. They will be executed prior to the current target if required.

```
<target name="exec" depends="compile">
  <java classname="compiler.Compiler">
    <classpath>
      <pathelement location="bin"/>
      <pathelement path="${classpath}"/>
    </classpath>
  </java>
</target>
```

The *java* task will execute the specified java class at the given *location* using *classpath*. When a name is prefixed by a dollar sign it refers to a *property* that is either defined in the build file or by the system.

System properties

os.name	Operating system name
os.arch	Operating system architecture
os.version	Operating system version
file.separator	File separator ('/' on UNIX)
path.separator	Path separator (':' on UNIX)
line.separator	Line separator ('\n' on UNIX)
java.class.path	Java class path
user.name	User's account name
user.home	User's home directory
user.dir	User's current working directory
...	

build.xml for CalcScan

```
<project name="Scanner" default="build">
  <property name="parser.program"
     value="test.TestScanner" />
  <property name="lib" value="lib" />
  <property name="parser.name" value="Parser" />
  <property name="specification.directory"
      value="specification" />
  <property name="parser.package" value="parser" />
  <property name="parser.directory"
     value="src/${parser.package}" />
```


build.xml for CalcScan

```
<target name="build">
  <mkdir dir="${parser.directory}" />
  <javacc target=
   "${specification.directory}/${parser.name}.jj"
      outputdirectory="${parser.directory}"
      javacchome="${lib}"
      static="false" />
  <antcall target="compile" />
</target>

<target name="compile" unless="nocompile">
  <mkdir dir="bin"/>
  <javac srcdir="src" destdir="bin"
     classpath="${lib}/junit.jar" />
</target>
```


build.xml for CalcScan

```
<target name="output"
    depends="build"
    description="Creates a result file in the project direc
  <input message="Target name" addproperty="target.name" />
  <java classpath="bin"
      classname="${parser.program}"
      output="${target.name}.res">
    <arg value="data/${target.name}" />
  </java>
  <echo>Creating ${target.name}.res</echo>
</target>
```