
Lecture Notes on Ant (COMP 303)

- These slides extracted from material at:
 - <https://www6.software.ibm.com/developerworks/education/j-apan/j-apan-ltr.pdf> (IBM Developer Works: Ant Tutorial)
 - <http://ant.apache.org/manual/> (Ant Users' Manual)
- Slides compiled by Laurie Hendren, McGill University.

[next](#) ... [Slide 1] ...

Ant is a Java-based build tool

- Original author, James Duncan, called it *Another Neat Tool*
- Helps automate the build process.
- Has many built-in tasks (javac, javadoc, junit,)
- Also allows you to define custom tasks.
- Allows you to define dependencies between tasks.
- Has won numerous awards.
- Has become the *de facto* standard for building open source Java projects.

[previous](#) | [start](#) | [next](#) ... [Slide 2] ...

Ant basics

- Ant uses build specification files written in XML.
- There are a defined set of XML elements that Ant understands.
- Ant can be extended.
- The default build file is called `build.xml`.
- One runs ant using the command:

```
ant (uses the default build.xml file, executes the default target in that file)
ant targetname (build.xml, specific target)
ant -buildfile buildfilename targetname (use a different build file, specific target)
```

[previous](#) | [start](#) | [next](#) ... [Slide 3] ...

A minimal build.xml file

```
<?xml version="1.0"?>
<project default="init">
  <target name="init">
    <target/>
  </target>
</project/>
```

- Must have a default target.
- Default target must be defined.
- Could also say `<target name="init" />`

[previous](#) | [start](#) | [next](#) ... [Slide 4] ...

Can add descriptions and comments

```
<xml version="1.0"?>
<project default="init" name="Project Argon">

<description>
  A simple project introducing the use of descriptive tags in Ant build files.
</description/>

<!-- XML comments can also be used -->
<target name="init" description="Initialize Argon database">
  <!-- perform initialization steps here -->
</target/>
</project/>
```

[previous](#) | [start](#) | [next](#) [Slide 5]

Can define properties

```
<property name="database-file" location="archive/databases/${metal}.db"/>
<property name="database-file" location="archive\databases\${metal}.db"/>
```

- Can use either kind of path separators, will work on both DOS/Windows and unix systems.
- Try to avoid DOS drive names "C: / " , not portable.
- Use relative pathnames when possible.

[previous](#) | [start](#) | [next](#) [Slide 6]

Can define dependencies

```
<target name="init"/>
...
<target name="preprocess" depends="init"/>
...
<target name="compile" depends="init,preprocess"/>
...
<target name="package" depends="compile"/>
```

- User can ask to start at any task target.
- `ant compile` will run `init` and `preprocess` if not already run, and then `compile`.

[previous](#) | [start](#) | [next](#) [Slide 7]

Let's look at a complete example

- Using the money example from the Junit Samples directory.
-

```
build.xml
src/
  money/
    IMoney.java
    Money.java
    MoneyBag.java
    MoneyTest.java
classes/
lib/
doc/
```

- compiled classes to go in `classes/` directory
- generated `.jar` files to go in `lib/` directory
- generated javadoc html to go in `doc/` directory

[previous](#) | [start](#) [Slide 8]