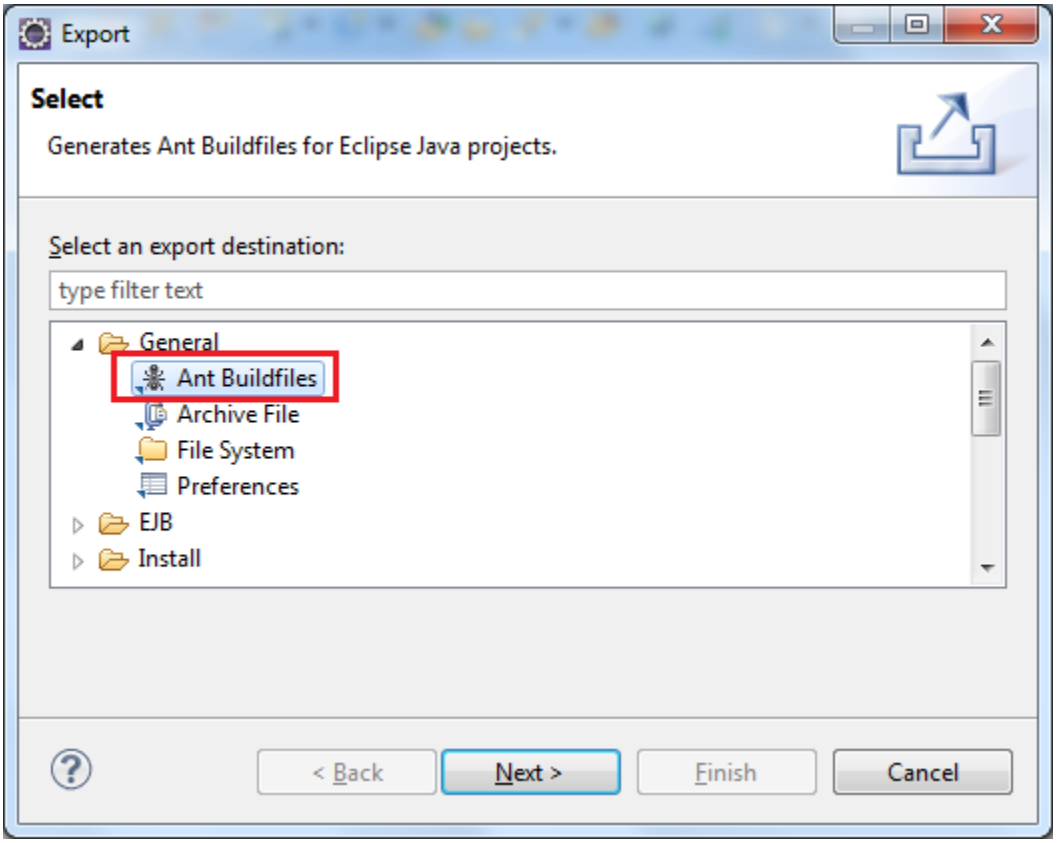
How to Create ANT Build File in Eclipse

Sometimes we need to build our Java projects from outside Eclipse IDE, by executing an Ant build script from command line. For example, when we made some very little changes to the project and need to re-build it without opening Eclipse (which may take times and huge memory for the bulky IDE while we are busy on other stuffs). Another typical case is bug-fixing, re-configuration and re-deployment in a production environment (i.e. a web application running on a server) in which we cannot use an IDE to re-built the project. Having only project's source code with all necessary jar files, the only missing thing is an Ant build file from which we can build the project from command line independently of Eclipse.

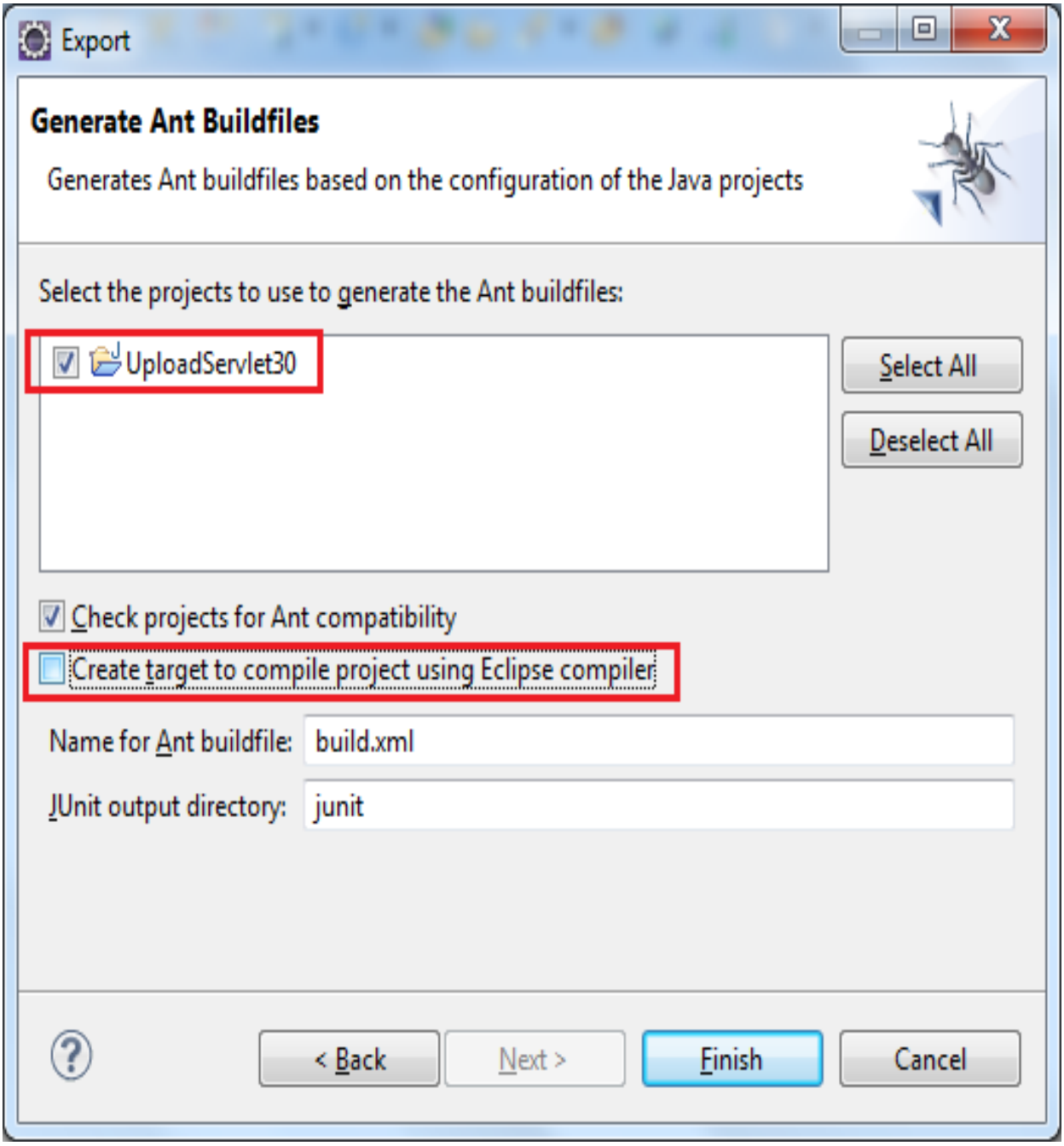
Although the build system in Eclipse does not based on Ant, Eclipse offers a function that lets us create Ant build script for the project manually.

Suppose we are working on a Java web application project named "UploadServlet30" in Eclipse. To create Ant build file for this project, follow these steps:

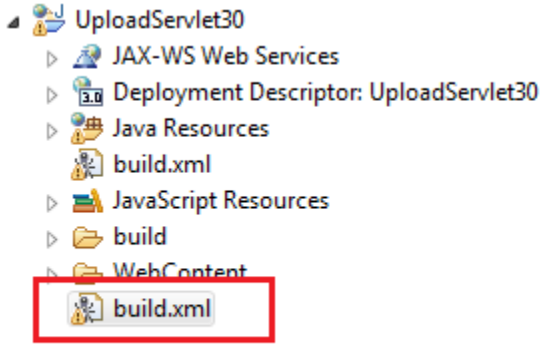
- Select **File > Export** from main menu (or right click on the project name and select **Export > Export…**).
- In the *Export* dialog, select **General > Ant Buildfiles** as follows:



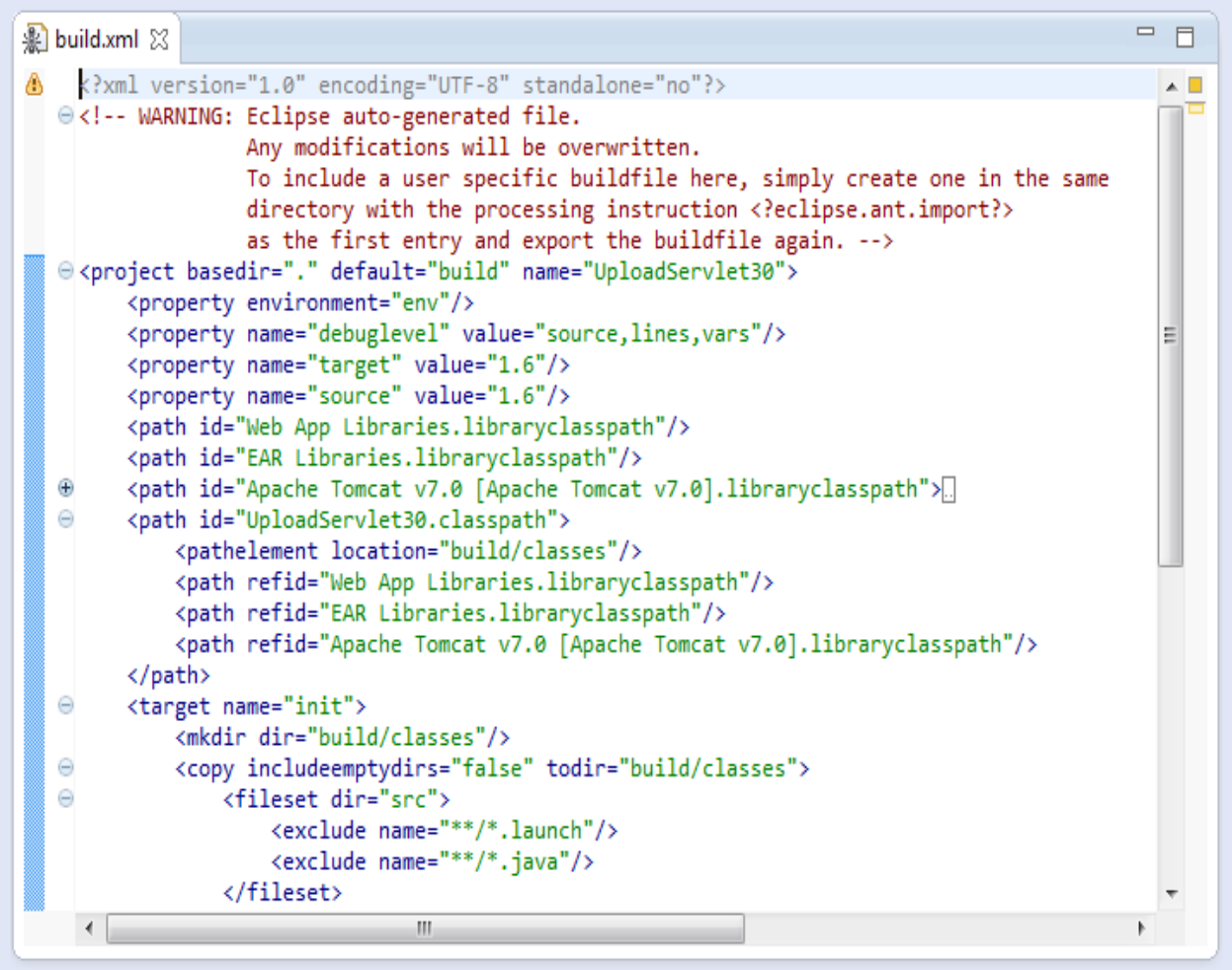
- Click **Next**. In the *Generate Ant Buildfiles*screen:
    - Check the project in list.
    - Uncheck the option "*Create target to compile project using Eclipse compiler*" because we want to create a build file which is independent of Eclipse.
    - Leave the *Name for Ant buildfile* as default: build.xml



- Click **Finish**, Eclipse will generate the **build.xml** file under project's directory as follows:



Double click on the build.xml file to open its content in Ant editor:



We can see that Eclipse generated a complete Ant build file with all libraries referenced by the project and standard build targets such as init, clean, build, etc. However, Eclipse did not generate a target to create WAR file for this web application, so we have to add it manually. Add the following code snippet at the end of the build.xml file, just above the closing tag </project>:

```
<target name="war" description="Bundles the application as a WAR file"
      depends="clean, build">
    <mkdir dir="WebContent/WEB-INF/classes"/>

    <copy includeemptydirs="false" todir="WebContent/WEB-INF/classes">
        <fileset dir="build/classes">
            <include name="**/*.class"/>
        </fileset>
    </copy>

    <war destfile="UploadServlet30.war" basedir="WebContent" needxmlfile="false">
    </war>
</target>
```

Save the file. Now switch to a command line prompt, change current directory to the project's directory, and type the following command to compile and build WAR file for the project:

C:/Go/To/Project/Parent/Directory> ant war

Output:

```
@)ant war
Buildfile: e:\Java\JavaEE\UploadServlet30\build.xml

clean:
   [delete] Deleting directory e:\Java\JavaEE\UploadServlet30\build\classes

build-subprojects:

init:
    [mkdir] Created dir: e:\Java\JavaEE\UploadServlet30\build\classes

build-project:
     [echo] UploadServlet30: e:\Java\JavaEE\UploadServlet30\build.xml
    [javac] Compiling 1 source file to e:\Java\JavaEE\UploadServlet30\build\classes
    [javac] warning: [options] bootstrap class path not set in conjunction with -source 1.6
    [javac] 1 warning

build:

war:
     [copy] Copying 1 file to e:\Java\JavaEE\UploadServlet30\WebContent\WEB-INF\classes
      [war] Building war: e:\Java\JavaEE\UploadServlet30\UploadServlet30.war

BUILD SUCCESSFUL
Total time: 0 seconds
```

NOTE: You must have Ant installed and configured properly (Environments variable for ant must be set) to execute the above command. This Ant installation is separate from the one bundled with Eclipse.