ANT - BUILD DOCUMENTATION

Documentation is a must in any project. Documentation plays a great role in the maintenance of a project. Java makes documentation easier by the use of the in-built **javadoc** tool. Ant makes it even easier by generating the documentation on demand.

As you know, the javadoc tool is highly flexible and allows a number of configuration options. Ant exposes these configuration options via the javadoc task. If you are unfamiliar with javadocs, we suggest that you start with this Java Documentation Tutorial.

The following section lists the most commonly used javadoc options that are used in Ant.

Attributes

Source can be specified using **sourcepath**, **sourcepathref** or **sourcefiles**.

- **sourcepath** is used to point to the folder of the source files *e. g. srcfolder*.
- **sourcepathref** is used to refer a path that is referenced by the path attribute *e. g, delegates. src. dir*.
- **sourcefiles** is used when you want to specify the individual files as a comma separated list.

Destination path is specified using the **destdir** folder *e. gbuild. dir*.

You could filter the **javadoc** task by specifying the package names to be included. This is achieved by using the **packagenames** attribute, a comma separated list of package files.

You could filter the javadoc process to show only the public, private, package, or protected classes and members. This is achieved by using the **private**, **public**, **package** and **protected** attributes.

You could also tell the javadoc task to include the author and version information using the respective attributes.

You could also group the packages together using the **group** attribute, so that it becomes easy to navigate.

Putting it all together

Let us continue our theme of the **Hello world** Fax application. Let us add a documentation target to our Fax application project.

Given below is an example javadoc task used in our project. In this example, we have specified the javadoc to use the **src.dir** as the source directory, and **doc** as the target.

We have also customized the window title, the header, and the footer information that appear on the java documentation pages.

Also, we have created three groups:

- one for the utility classes in our source folder,
- one for the user interfaces classes, and
- one for the database related classes.

You may notice that the data package group has two packages -– faxapp.entity and faxapp.dao.

```
<target name = "generate-javadoc">
   <javadoc packagenames="faxapp.*" sourcepath="${src.dir}"
      destdir = "doc" version = "true" windowtitle = "Fax Application">

      <doctitle><![CDATA[= Fax Application =]]></doctitle>
      <bottom>
         <![CDATA[Copyright © 2011. All Rights Reserved.]]>
      </bottom>

      <group title = "util packages" packages = "faxapp.util.*"/>
      <group title = "web packages" packages = "faxapp.web.*"/>
      <group title = "data packages" packages = "faxapp.entity.*:faxapp.dao.*"/>
   </javadoc>

   <echo message = "java doc has been generated!" />
</target>
```

Let us execute the javadoc Ant task. It generates and places the java documentation files in the doc folder.

When the **javadoc target** is executed, it produces the following outcome:

```
C:\>ant generate-javadoc
Buildfile: C:\build.xml

java doc has been generated!

BUILD SUCCESSFUL
Total time: 10.63 second
```

The java documentation files are now present in the **doc** folder.

Typically, the javadoc files are generated as a part of the release or package targets.