Ant

Introducción

Ant es una herramienta para facilitar las tareas de compilación y construcción de un proyecto en Java. Es similar a la herramienta Make, pero es independiente del sistema operativo, dado que está desarrollado en Java.

Instalación

Esta aplicación no se instala, directamente se descomprime en un directorio y se la utiliza. La distribución más reciente se puede conseguir desde la página http://ant.apache.org/. Una vez descomprimida, los ejecutables se encuentran en el directorio ***bin***.

Para utilizarla en el laboratorio, se la puede bajar y descomprimir en el *home* del usuario actual, por ejemplo en:

```
/home/user/programas/apache-ant-1.7.0
```

Luego, es necesario agregar el directorio ***bin*** al *path* del sistema. Para hacer esto, se debe editar el archivo ***.bash_profile*** ubicado en el *home* del usuario actual (si no existe crearlo) y agregar la línea:

```
PATH=$PATH:/home/user/programas/apache-ant-1.7.0/bin
```

Luego, abrir otra terminal y verificar que desde cualquier otro directorio se pueda ejecutar el comando ***ant***.

Uso de la herramienta

Para utilizar ***Ant*** en un proyecto se debe escribir un *buildfile*, que es un archivo que describe cómo se debe compilar dicho proyecto (similar al *makefile*). Este archivo debe llamarse *build.xml* y estar ubicado en la raíz del proyecto. Es un archivo de texto en formato xml, que tiene el siguiente aspecto:

```
<project name="MyProject" default="C">
  <target name="A">
    ...
  </target>
  <target name="B" depends="A">
    ...
  </target>
  <target name="C" depends="B">
    ...
  </target>
</project>
```

La raíz del documento es el tag ***project***, que define el nombre del proyecto, y contiene una lista de ***targets***. Un ***target*** es un objetivo (por ejemplo compilar, armar un archivo ***jar***, correr los tests, etc.) que está formado por una o más tareas; y a su vez puede depender de uno o más *targets*. El proyecto define un *target default*, que es el predeterminado para construir toda la aplicación.

Una vez creado este archivo, si se ejecuta el comando ***ant***, entonces se busca el *target default* y se ejecutan sus tareas. Si este ***target*** depende de otros, primero se ejecutan estos y luego el ***target*** en cuestión (y así sucesivamente si siguieran habiendo dependencias).

Si se quiere ejecutar algún *target* en particular, entonces se debe ejecutar el comando ***ant*** seguido del nombre del *target*. En el ejemplo anterior, si se ejecuta "***ant C***", entonces se ejecuta el *target* C, pero como este depende del B, que a su vez depende del A, se ejecutan los *targets* A, B y C (en ese orden).

Un target contiene un conjunto de tareas, que son descriptas utilizando tags específicos para cada caso.

Tareas de uso común

Javac

Es utilizado para compilar la aplicación. El parámetro *srcdir* indica el directorio donde se encuentran los archivos java de la aplicación. El parámetro *destdir* indica en directorio en donde se deben dejar los archivos compilados (.class). Esta tarea compila recursivamente en todos los subdirectorios que encuentre, respetando la estructura de paquetes, y crea esta misma estructura en el directorio destino.

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

Jar

Es utilizado para generar un archivo ***jar***. El parámetro *basedir* indica el directorio a partir del cual se debe tomar el contenido a incluir en el ***jar***. El parámetro *destfile* indica el nombre del archivo a crear.

```
<jar destfile="proyecto.jar" basedir="bin" />
```

Si se desea agregar un *manifest* al ***jar*** (por ejemplo para indicar la clase principal), se lo puede hacer a través del tag ***manifest***:

```
<jar destfile="proyecto.jar" basedir="bin">
   <manifest>
      <attribute name="Main-Class" value="paquete.ClasePrincipal"/>
   </manifest>
</jar>
```

Javadoc

Es utilizado para crear la documentación de la aplicación. El parámetro *sourcepath* indica el directorio donde se encuentran los códigos fuente. El parámetro *destdir* indica el directorio donde se debe crear el sitio con la documentación.

```
<javadoc sourcepath="src" destdir="doc" />
```

JUnit

Es utilizado para ejecutar testeos de unidad realizados con ***JUnit***.

```
<junit>
  <test name="tests.SampleTest1" />
  <test name="tests.SampleTest2" />
</junit>
```

Por default si algún test falla no se informa el motivo. Si se quiere obtener una salida más detallada por consola, se debe agregar el tag ***formatter*** como se muestra a continuación. Además, para que funcione correctamente, el ***jar*** de ***JUnit*** y las clases de testeo deben estar en el *classpath*. De no estarlo, es necesario agregar explícitamente su ubicación. En el ejemplo siguiente se asume que las clases compiladas se encuentran en el directorio ***bin*** y que el ***jar*** de ***JUnit*** se encuentra en el directorio ***lib***:

```
<junit>
  <formatter type="brief" usefile="false"/>
  <classpath path="lib/junit-4.7.jar;bin" />

  <test name="tests.SampleTest1" />
  <test name="tests.SampleTest2" />
</junit>
```

Ejemplo

A continuación se muestra un *buildfile* de ejemplo con estas tareas básicas. Se suele agregar un *target* llamado ***clean***, que elimine todos los archivos binarios creados por *targets* anteriores.

```
<project name="Sample" default="dist" basedir=".">

  <!-- Compila el proyecto, deja los archivos class en el directorio bin -->
  <target name="compile">
    <mkdir dir="bin"/>
    <javac srcdir="src" destdir="bin" classpath="lib/junit-4.7.jar"/>
  </target>

  <!-- Crea la documentacion en formato Javadoc, en el directorio doc. -->
  <target name="doc" depends="compile">
    <javadoc sourcepath="src" destdir="doc" classpath="lib/junit-4.7.jar"/>
  </target>

  <!-- Ejecuta los tests. -->
  <target name="tests" depends="compile">
    <junit>
      <formatter type="brief" usefile="false"/>
      <classpath path="lib/junit-4.7.jar;bin" />

      <test name="tests.SampleTest1" />
      <test name="tests.SampleTest2" />
    </junit>
  </target>

  <!-- Crea el jar ejecutable con todo el proyecto compilado. -->
  <target name="dist" depends="compile, doc, tests">
    <jar destfile="sample.jar" basedir="bin">
      <manifest>
        <attribute name="Main-Class" value="sample.SampleApp"/>
      </manifest>
    </jar>
  </target>

  <!-- Borra todos los archivos generados luego de compilar. -->
  <target name="clean">
    <delete dir="bin"/>
    <delete dir="doc" />
    <delete file="sample.jar" />
  </target>

</project>
```