

Ejercicios de Automatización con Ant

Índice

1 Ant básico.....	2
2 Ant avanzado (*).....	2

1. Ant básico

Vamos a trabajar con una aplicación de prueba llamada *jhd-sesion2*. El directorio de desarrollo de la aplicación tiene la siguiente estructura de subdirectorios:

<code>src</code>	Código fuente de la aplicación.
<code>bin</code>	Directorio donde generaremos la aplicación compilada.
<code>dist</code>	Directorio donde generaremos el paquete JAR para distribuir la aplicación.
<code>docs</code>	Directorio con la documentación Javadoc de nuestra aplicación.

Se pide:

- a) Escribir un fichero `build.xml` básico para este proyecto, con un objetivo `compile` que nos permita compilar la aplicación, guardando las clases compiladas en el directorio `bin`. Se debe intentar parametrizar este fichero en la medida de lo posible.
- b) Añadir un objetivo `run` que nos permita ejecutar la clase principal de la aplicación (`es.ua.jtech.jhd.sesion2.ejerc1.holamundo.Principal`). Este objetivo deberá depender de `compile`.
- c) Añadir un objetivo `package` que genere un paquete JAR con la aplicación en el directorio `dist`. Indicar en este paquete la clase principal de nuestra aplicación en los atributos del MANIFEST para hacerlo autoejecutable. Este objetivo dependerá de `compile` también.
- d) Modificar el objetivo `run` para que ejecute la aplicación desde el JAR, en lugar de hacerlo desde el directorio de clases compiladas. En este caso `run` deberá depender de `package`.
- e) Añadir un objetivo `doc` que genere la documentación de la aplicación en formato Javadoc en el directorio `docs`.
- f) Añadir un objetivo `clean`, que limpie todo lo que se ha construido de la aplicación. Debe borrar todo el contenido del directorio `bin` y `dist`.

2. Ant avanzado (*)

Ahora vamos a trabajar con una versión extendida de la aplicación anterior, en la que vamos a incorporar recursos (imágenes) y vamos a necesitar utilizar una librería externa. Estos nuevos elementos estarán en los siguientes directorios:

<code>resources</code>	Recursos de la aplicación.
<code>lib</code>	Librerías de la aplicación.

Lo primero que debemos hacer, es descomentar de las clases `Principal` y `Ventana` todos aquellos bloques de código indicados. Una vez hecho esto, coger el fichero `build.xml` realizado en el ejercicio anterior y extenderlo con las siguientes características:

a) Modificar el *classpath* de compilación y de ejecución en el fichero `build.xml` para que incorpore todas las librerías `*.jar` que se encuentren dentro del directorio `lib`.

NOTA: Al necesitar introducir más elementos en el *classpath*, ya no será posible utilizar un JAR autoejecutable. Modificar el objetivo `run` para que ejecute a partir del fichero JAR, pero simplemente introduciendo dicho fichero en el *classpath* y especificando la clase principal que queremos ejecutar. De esta forma, pondremos en el *classpath* el fichero JAR en el que hemos empaquetado la aplicación en el directorio `dist`, y todos aquellos ficheros JAR que se encuentren en el directorio `lib`.

NOTA: Es importante poner en la tarea `java` el atributo `fork="true"` para que la aplicación pueda funcionar correctamente. De no hacerlo, puede que debido a conflictos con Eclipse se produzcan errores de ejecución.

b) Antes de generar el paquete de la aplicación, en el objetivo `package`, copiar todos los recursos de `resources` a `bin`, para que estos recursos sean añadidos a este paquete. Tras hacer esto, en la ventana de la aplicación deberemos ver una imagen.

