Paquete. Problemas de compilación y otras notas

Vamos a analizar los problemas de compilación que pueden surgir cuando se utilizan paquetes. Para ello, supongamos que tenemos un CLASSPATH definido como sigue:

```
c:> set CLASSPATH=.;c:\java\util
```

También vamos a suponer que tenemos un paquete llamado `lista`. Este paquete tiene un fichero `Lista.java` que incluye dos clases `Lista` y `NodoLista` y otro `ListaIter.java` que incluye una clase `ListaIter.` Los dos ficheros del paquete comienzan con la línea `package lista;`

Además disponemos de un fichero `EjLista.java` con una clase `EjLista` que utiliza las clases del paquete anterior. Este fichero incluye la línea `import lista.*;` antes de la definición de ninguna clase o interface.

Compilación de los ficheros pertenecientes a un paquete

Para compilar el paquete, debemos colocar los ficheros pertenecientes al paquete en un directorio con el nombre del paquete. En nuestro caso, `lista` (cuidado con las mayúsculas). Da igual donde esté el directorio. Por ejemplo, supongamos que lo colocamos en:

```
c:\curso2000\java\sesion3\lista\Lista.java
c:\curso2000\java\sesion3\lista\ListaIter.java
```

Entonces, la compilación la debemos hacer desde la raíz del paquete indicando el camino:

```
c:\curso2000\java\sesion3> javac lista/Lista.java
c:\curso2000\java\sesion3> javac lista/ListaIter.java
```

Nótese que el separador es `/` y no `\`. Si no hay errores de compilación, aparecerán en el directorio `c:\curso2000\java\sesion3\lista` los ficheros `Lista.class`, `NodoLista.class`, `ListaIter.class`.

Ubicación del paquete dentro de un directorio indicado por **`CLASSPATH`**

Las clases (`.class`) del paquete creado deben ubicarse adecuadamente para que sea accesible a través de `CLASSPATH`. Según está definido `CLASSPATH`, hay dos posibilidades:

- Dejarlas donde están, esto es, en el directorio `c:\curso2000\java\sesion3\lista>`. Esto obligaría a que cualquier clase que utilice al paquete tenga que colocarse en el directorio
  `c:\curso2000\java\sesion3>`
  Así, esta clase encuentra al paquete pues el directorio actual (`.`) se encuentra en `CLASSPATH` y como subdirectorio (o paquete) aparece `lista` con las clases.

- Colocar los ficheros `.class` del paquete en el directorio
  `c:\java\util\lista>`
  Ahora cualquier clase que utilice al paquete puede colocarse donde quiera.
  Así, esta clase encuentra al paquete pues el directorio `c:\java\util` se encuentra en `CLASSPATH` y como subdirectorio (o paquete) aparece `lista` con las clases.

Algunas consideraciones importantes

Alternativas de importación

Existe una diferencia entre una importación cualificada del paquete

```
import lista.*;
```

y una importación cualificada de las clases necesarias del paquete

```
import lista.Lista;
import lista.ListaIter;
```

En el primer caso, estamos indicando que importe todas las clases del paquete `lista` **que no hayan podido resolverse** una vez observadas las que hay en el directorio actual. Esto quiere decir que si en el directorio actual se encuentra una clase llamada `Lista`, se utilizará ésta en lugar de la del paquete.

En el segundo caso, estamos indicando **que importe y utilice** las clases `Lista` y `ListaIter` aún cuando en el mismo directorio de trabajo existieran esas clases.

Ficheros que utilizan paquetes

Los paquetes y los ficheros que lo usan no deben colocarse en el mismo directorio. Lo normal es disponer de lugares concretos para colocar paquetes y otros donde se colocan las aplicaciones. En cualquier caso, si se colocan juntos, cualquier importación debe ser cualificada de clase para que se utilicen las clases del paquete y no las que hay en el directorio aún cuando sean las mismas.
Si no se hace así, se intentará utilizar una clase encontrada en el mismo directorio pero que incluye la palabra `package` por lo que producirá error.

Importación cualificada

Es posible utilizar clases de un paquete sin necesidad de escribir la cláusula de importación. Para ello, es suficiente con hacer un uso cualificado de la clase.
Por ejemplo, un uso cualificado de la clase `Lista` del paquete `lista` en otra clase que no la importa explícitamente puede ser

```
...
lista.Lista li = new lista.Lista();
...
```

Duplicación de nombres

Supongamos ahora, que disponemos de dos clases `Lista`. La que proporciona el paquete `lista` y otra que tenemos en el mismo directorio de trabajo. Podemos utilizar ambas listas si no importamos cualificadamente la clase y cualificamos su uso

```
class PL {
     static public void main(String [] args) {
        lista.Lista li = new lista.Lista();       // (1)
        Lista oli      = new Lista();             // (2)
        ...
```

En (1) estamos utilizando la `Lista` del paquete y en (2) la que se encuentra en nuestro directorio.
Del mismo modo, podemos utilizar dos clase con el mismo nombre pero de paquetes distintos simplemente cualificando su uso.