Algunas cosas sobre uso de paquetes en Java

Los paquetes son el mecanismo que usa Java **para facilitar la modularidad y la organización del código fuente, sobre todo cuando trabajamos con aplicaciones grandes que cuentan con un gran números de clases, subclases, interfaces, librerías, etc**.

Un paquete puede contener una o varias clases y otros elementos.

Cuando trabajamos con varios paquetes, **cada fichero .java** o módulo de código, **comenzará con la palabra clave** `package` seguida del nombre del paquete al que pertenece.

Ejemplo : package Producto;

Para utilizar los elementos de un paquete desde otro **es necesario importar** el primero en el código fuente en curso, usando para ello la sentencia **`import`.**

Un paquete puede contener, además elementos como clases e interfaces, otros paquetes, **dando lugar a estructuras jerárquicas**.

Para referirnos a estos **subpaquetes, paquetes contenidos en otros**, usaremos el identificador del paquete principal seguido de un punto y el nombre del subpaquete. Si existieran más niveles en la jerarquía estos se agregarían al identificador separados por puntos, formando así el nombre completo del paquete.

Comentar además que, esta jerarquía de paquetes y subpaquetes coincidirá con una estructura jerárquica de carpetas en nuestro sistema de almacenamiento cuyos nombres serán los mismos que los paquetes asociados.

Importación de paquetes

La cláusula `import` puede utilizarse para importar todo el paquete completo o sólo otro subpaquete si lo tuviese, o un elemento concreto.

¿Cómo especificar qué parte del paquete queremos importar?

Poniendo el nombre de este seguido de un punto y el identificador de dicho elemento. Por ejemplo, para importar la clase `Math` del paquete `java.lang`, pudiendo así acceder a la constante `PI` y las funciones matemáticas que aporta, bastaría con la siguiente línea:

```
import java.lang.Math;
```

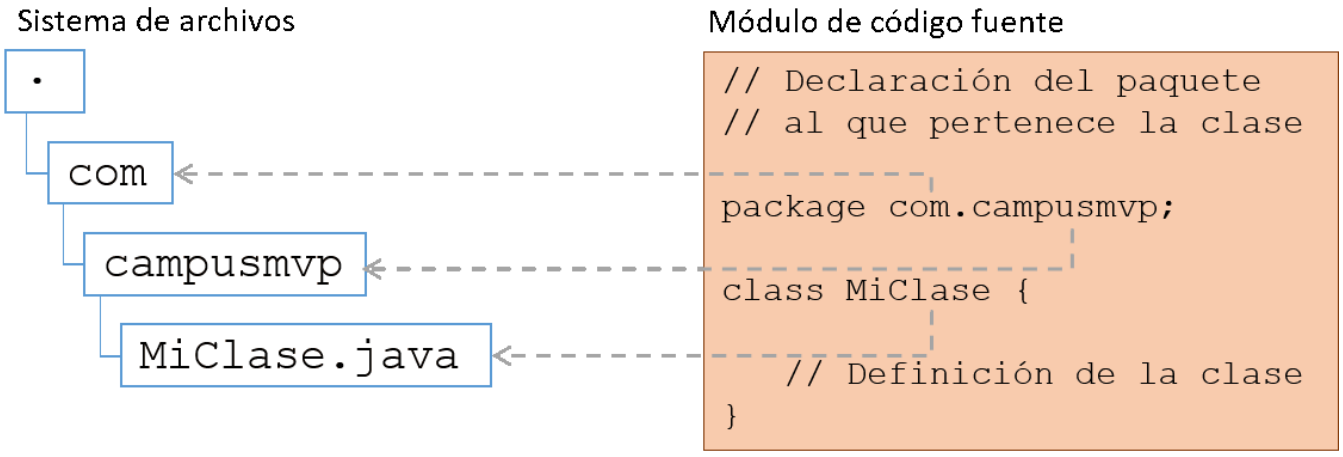
Es habitual que al importar un paquete nos interesen muchas de las clases definidas en él. En este caso podríamos importarlas todas de una vez, así:

```
import java.lang.*;
```

Relación entre clases, paquetes y el sistema de archivos

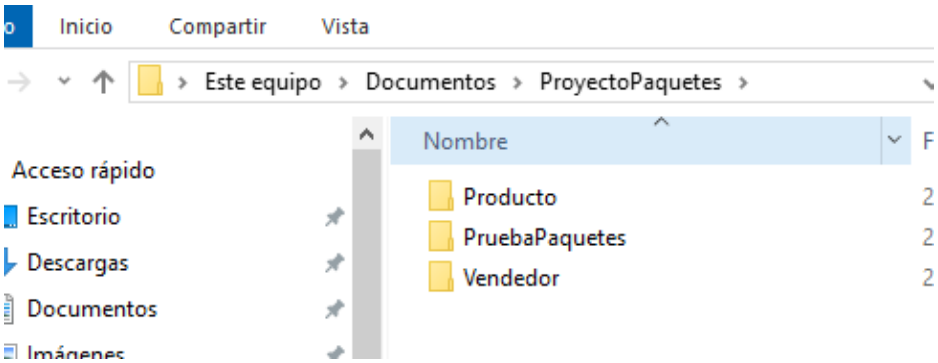
En Java existe una estrecha relación entre **las definiciones de paquetes y clases y el sistema de archivos local**, en el que se almacenan los ficheros conteniendo el código. Es un hecho que puede pasarnos totalmente inadvertido al trabajar con un IDE como el de NetBeans, ya que sus opciones se encargan de ocultar estos detalles

El **nombre del paquete debe coincidir además con el nombre de la carpeta donde se alojan los módulos de código** contenidos en el paquete. Los nombres de paquete pueden ser compuestos, usándose el punto como separador de las diferentes porciones del identificador. Cada parte identificaría a un subdirectorio, según se muestra en el siguiente esquema:



Al trabajar desde la línea de comandos, usando directamente el JDK, hemos de tener en cuenta que tanto la compilación como la ejecución verificarán que se **existe correspondencia entre el nombre de paquete y el nombre del directorio que la contiene (**del mismo modo que comprueba que el fichero .java se llama igual que la clase que contiene). Si esta no se cumple en algún punto, como ocurre en la parte inferior de la siguiente imagen, se obtiene un error.

Si por ejemplo, yo tengo mi proyecto al que he llamado ProyectoPaquetes dentro de la carpeta ProyectoPaquetes que cuenta una estructura de carpetas como siguiente:



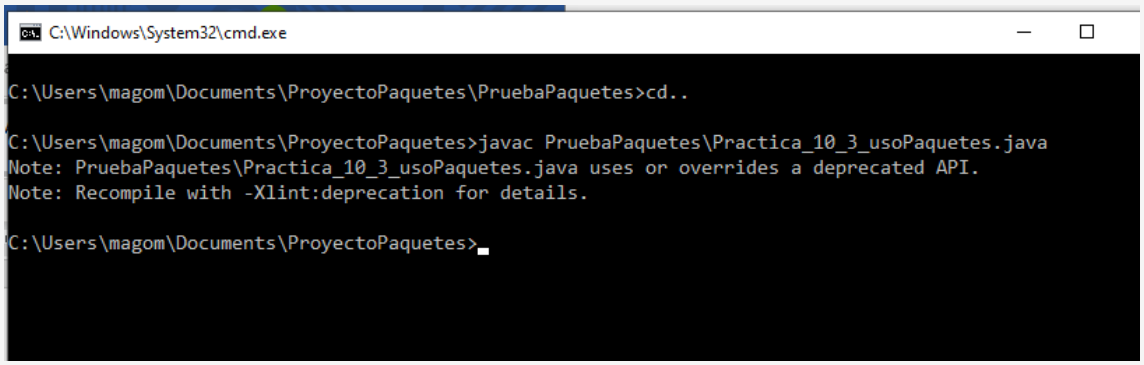
En cada carpeta tengo una clase distinta.

La clase que contiene el método main() de mi proyecto está en la carpeta PruebaPaquetes y tiene el nombre Practica_10_3_usoPaquetes.

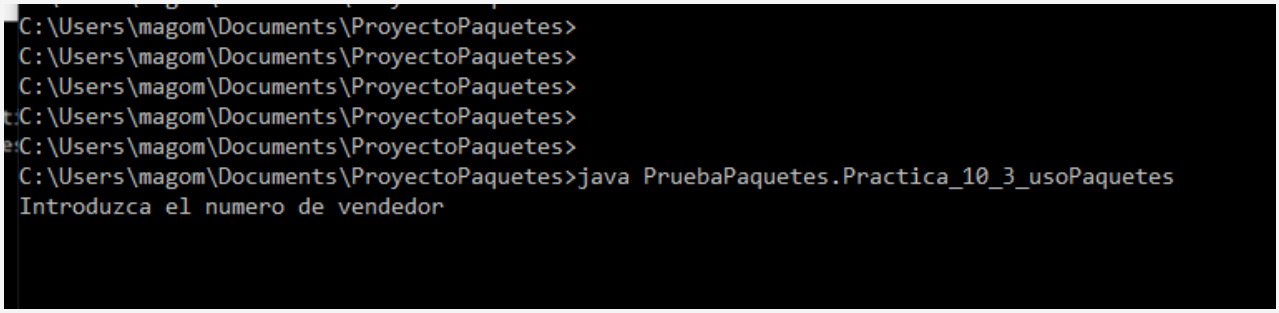
¿Cómo podría yo compilar y ejecutar mi aplicación desde JDK?

Compilar

Debemos situarnos en la consola dentro del directorio padre de PruebaPaquetes.



Ejecutar

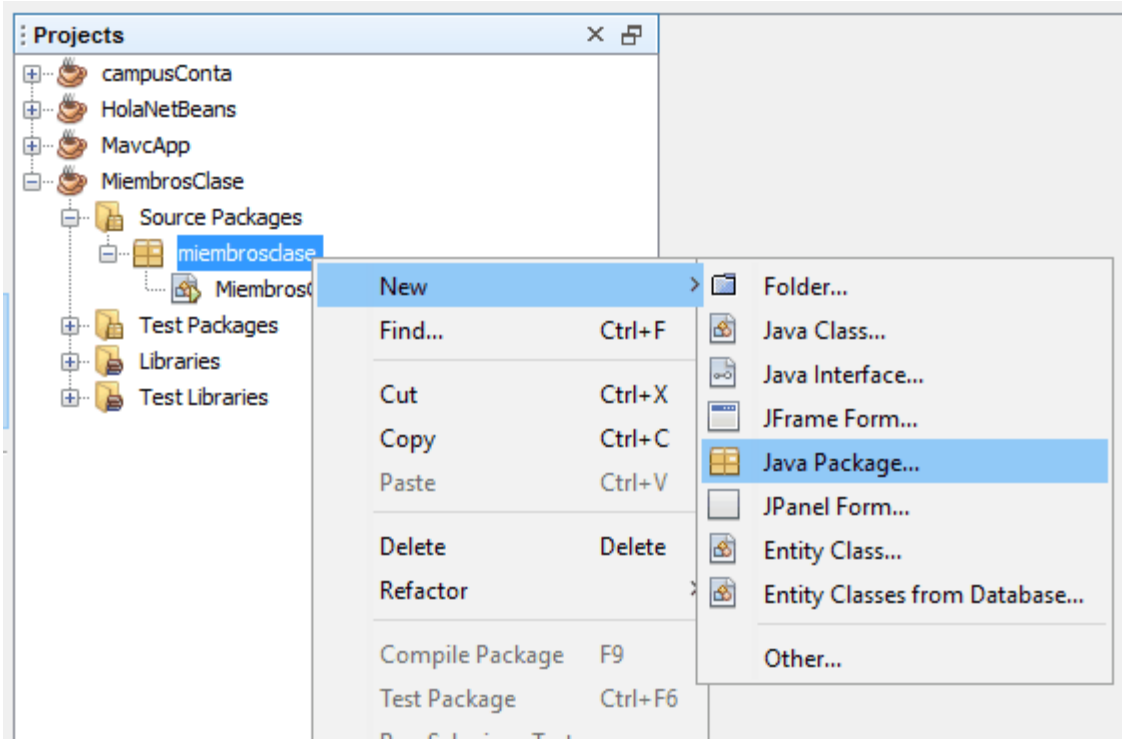


Nota: la estructura de directorios de un paquete puede almacenarse completa, una vez se compila el proyecto, en un archivo JAR. Este contendría asimismo los archivos de clase de cada módulo de código.

A continuación puedes ver un **vídeo práctico explicativo** de todo lo anterior, que está extraído directamente del [curso de programación con Java] de campusMVP:

Creación de paquetes usando NetBeans

Un paquete Java se genera sencillamente incluyendo la palabra clave `package` al inicio de los módulos de código en los que se definen las clases que formarán parte del mismo. Trabajando en un proyecto con NetBeans, comprobaremos que en la ventana `Projects` los paquetes se representan con un icono específico y actúan como nodos contenedores, alojando los módulos `.java` con el código fuente. El menú contextual del proyecto nos ofrece la opción `New>Java Package`, que será el que usemos habitualmente para crear un nuevo paquete:



Nota: cada vez que se crea un nuevo proyecto con NetBeans se propone la definición de un nuevo paquete, cuyo nombre sería el mismo del proyecto, donde se alojarían los módulos de código. En proyectos complejos, no obstante, puede ser necesaria la **creación de paquetes adicionales**.

Un paquete puede contener, además de definiciones de tipos como las clases e interfaces, otros paquetes, **dando lugar a estructuras jerárquicas de contenedores**. La denominación de los **subpaquetes, paquetes contenidos en otros**, se compondrán del identificador del contenedor seguido de un punto y el nombre del subpaquete. De existir niveles adicionales se agregarían los distintos identificadores separados por puntos, formando así el nombre completo del paquete.

El diagrama siguiente representa el contenido de un paquete llamado `campusmvp`, teóricamente correspondiente a un proyecto de aplicación de contabilidad. En ese paquete de primer nivel tenemos un subpaquete con utilidades, llamado `campusmvp.util`, no directamente vinculadas con la aplicación de contabilidad. El subpaquete `campusmvp.conta` contendría todos los elementos asociados a dicha aplicación, distribuidos a su vez en un tercer nivel de paquetes que contienen las clases correspondientes al modelo, la vista, etc., siguiendo al **patrón arquitectónico MVC**. Cada uno de los módulos de definición de clase, denominados `Cuenta.java`, `Movimiento.java`, `Vista.java`, etc., comenzarían con una cláusula `package` indicando **el nombre completo del paquete al que pertenecen**:

Nota: todas las clases que pertenecen a un mismo paquete comparten un ámbito común, al que pertenecerán aquellos miembros que no especifiquen explícitamente otro tipo de visibilidad.

Fuente: https://www.campusmvp.es/recursos/post/paquetes-en-java-que-son-para-que-se-utilizan-y-como-se-usan.aspx