Tema 8: Bases de datos orientadas a objetos

Secuenciación

Como ya sabemos, se define la persistencia de objetos al hecho de guardar toda la información que contiene un objeto manteniendo toda su estructura, dentro de una memoria secundaria. Para ello hay que usar o bien ficheros o bien bases de datos. En esta unidad se tratará como hacerlo usando bases de datos.

Base de datos orientada a objetos

Una base de datos orientada a objeto (BDOO) se caracteriza porque la información que guarda - como su propio nombre indica - son objetos, conservando toda su estructura de atributos y métodos.

Sin embargo, en una base de datos relacional había que extraer los datos que contenía un objeto para poder guardarlos en las diferentes tablas o al revés.

Gracias al uso de las bases de datos orientadas a objetos, es fácil guardar los datos que se manejan en un programa orientado a objetos, ya que conservan toda su estructura. De igual modo, a la hora de recuperarlos se recuperan en forma de objeto, permitiendo con ello hacer uso de los métodos de la clase a la que pertenece.

Características de las bases de datos orientadas a objetos:

- Guarda la información de los objetos, conservando toda la estructura de la clase a la que pertenece.
- En estas bases de datos, lo que identifica a cada objeto guardado en dicha base no son sus datos ni el nombre del objeto sino un identificador que se le da a cada objeto, el cual es único para cada objeto. Esto es lo que se conoce como OID (ObjectsIdentifier). Dicho identificador es independiente de los datos que contiene.
- En una misma base de datos se pueden almacenar objetos de diferentes tipos de clases.
- Permite guardar la información de objetos estructurados, es decir, objetos donde alguno de los datos es de tipo objeto. Este tipo de base de datos permite manejar esta información de forma muy sencilla.

Instalación del gestor de bases de datos

Hay muchos gestores de bases de datos para objetos (ODBMS, ObjectDatabase Management System). El que usaremos nosotros es el db4o. Sus siglas significan "Database 4(for) Objects", lo cual significa base de datos para objetos. Db4o permite almacenar objetos Java de forma directa y fácil.

Algunas razones para elegir este gestor:

- De código abierto y libre bajo la GPL. Es la principal clave del éxito de este gestor de base de datos.
- Hace que sea fácil la persistencia de objetos.
- Es capaz de almacenar cualquier tipo de objeto, aunque la clase a la que pertenezca sea muy compleja. El gestor es capaz de guardar la información de los objetos definidos con este tipo de clase con una sola línea de código.
- Tiene un alto rendimiento.
- La base de datos que crea db4o es un único fichero de extensión ".yap"
- Algo muy importante de este gestor es que para poder trabajar con dicho programa no hay que instalar nada. Sólo hay que añadir a nuestro proyecto un fichero jar que contiene dicho gestor.

Para usar dicho gestor lo que tenemos que hacer es simplemente agregar el fichero anteriormente indicado a nuestro proyecto a través de la biblioteca. Así, a partir de este momento, podremos usar todas las clases que tiene la librería, que permitirán gestionar una base de datos orientada a objetos.

Creación de bases de datos

La única instrucción necesaria para crear una base de datos orientada a objetos es ésta:

```
ObjectContainer bd = Db4oEmbedded.openFile("nombrebd");
```

Db4oEmbedded es una clase que pertenece a la librería del gestor db4o que hemos importado. Esa librería tiene un método **openFile** y dicho método permite crear y abrir una base de datos. Si la base de datos ya estuviese creada, se limitaría a abrirla. Exactamente, lo que hace este fichero es crear un fichero (si no existiera) con el nombre indicado y con extensión ".yap"; dicho fichero lo guarda en la raíz del proyecto. Este fichero representa la base de datos.

El método **OpenFile()** devuelve un objeto de tipo **ObjectContainer;** este objeto es el que va a permitir el manejo de la base de datos, debido a que dicha clase tiene métodos para poder llevar a cabo operaciones como insertar, borrar o consultar objetos de una base de datos.

Por ello, una vez abierta la base de datos, podremos insertar objetos. Para ello hay que utilizar el método **store** de la clase **ObjectContainer.** Dicho método requiere un parámetro que es el objeto que queremos guardar en la base de datos.

Es muy importante cerrar la base de datos cuando sepamos que ya no la vamos a usar. La instrucción para cerrar la base de datos es:

Nombre_objeto_ObjectContainer.close();

Debe recordarse que esta base de datos es un fichero con el nombre de que se le ha dado a dicha base de datos con extensión .yap y está guardado en la raíz del proyecto.

Mecanismos de consulta

QBE

Las consultas sencillas se hacen por QBE. QBE viene de QueryByExample, que significa "Consulta por ejemplo".

Su nombre se debe a la forma que tiene de recuperar los objetos que contiene la base de datos. Para ello utiliza el método **QueryByExample()** que tiene la clase **ObjectContainer**. En la llamada a dicho método se le enviará un objeto que servirá de ejemplo con el fin de indicar cuáles son los objetos que queremos recuperar de la base de datos; los objetos que recupere serán aquellos que sean del mismo tipo que el objeto que recibe y que tengan los mismos datos. Si en algún dato no queremos plantear ninguna condición, a dicho objeto ejemplo se le asignará un valor **null** si es un atributo de tipo referencia o el valor 0 si es un atributo de tipo primitivo.