



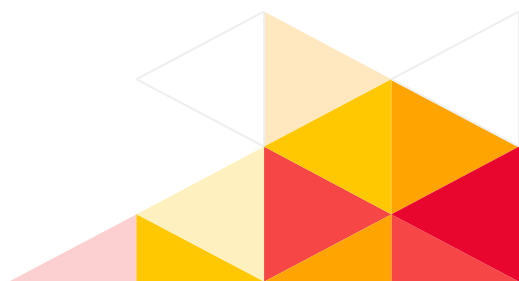
BDOO



Bases de datos orientadas a objetos

Las BDOO almacenan los datos en estructuras iguales a las utilizadas en POO, por lo que simplifican el proceso de persistencia de los datos de ese tipo de programas.

En la actualidad se utilizan distintos tipos de base de datos noSQL que no son propiamente orientadas a objetos






Neodatis

Es una BDOO muy simple que corre en Java, .Net, Android ...

Se almacena en un archivo local.

Lo cierto es que el proyecto lleva “parado” unos cuantos años, pero nos va a permitir vislumbrar cómo funciona este mundo de las BDOO.





Neodatis

- ◀ La unidad de persistencia es el objeto, no la tabla.
- ◀ Los objetos son persistidos tal y como aparecen en el lenguaje de programación, sin ninguna conversión.





Abrir conexión BD

La base de datos se almacena localmente. La declaramos e inicializamos con:

```
ODB miOdb =  
ODBFactory.open("baseDatos00.odb")
```


Si no existe se creará.





Guardar un objeto

Simplemente habrá que invocar sobre el objeto ODB el método `store (objeto)`





Consultas de lectura

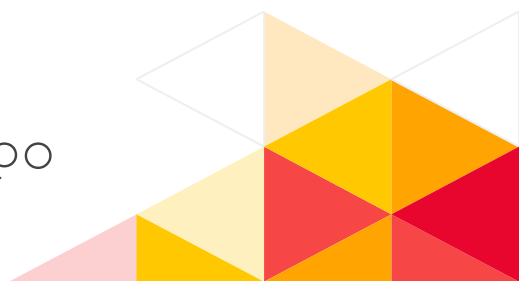
Para leer un objeto necesitamos montar una consulta `IQuery` a través de un nuevo objeto de la clase

```
CriteriaQuery(ClaseDelObjeto.class,  
Where.equal("NombreAtributo", valor))
```

Luego la ejecutamos mediante

```
miOdb.getObjects(consulta)
```

Lo devolverá en un objeto de tipo `Objects<clase>`





```
Deporte miDeporte = new Deporte( nombre: "Petanca");
Deporte otroDeporte = new Deporte( nombre: "Bolos");


ODB mi0db = ODBFactory.open( FileName: "ejemplo1.odb");

mi0db.store(miDeporte);

IQuery consulta = new CriteriaQuery(Deporte.class, Where.equal( attributeName: "nombre", value: "Petanca"));

Objects <Deporte> deportes = mi0db.getObjects(consulta);

while(deportes.hasNext())
    System.out.println(deportes.next());
mi0db.close();
```





Consultas de lectura

También podemos traernos todos los objetos de una clase concreta mediante

```
Objects <Deporte> deportes =  
miOdb.getObjects(Deporte.class);
```

Y luego lo recorremos con el típico

```
while(deportes.hasNext())  
    deportes.next();
```

