Tài liệu hướng dẫn

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

Nhóm 2 | Mẫu thiết kế hướng đối tượng

*Xây dựng database access management (DAM) framework*

Mục lục

[I. Hướng dẫn cài đặt: 2](#_Toc486798258)

[II. Hướng dẫn sử dụng: 2](#_Toc486798259)

# Hướng dẫn cài đặt:

Bước 1: tải các driver kết nối dữ liệu mà mình cần và add vào project

Bước 2: tải các thư viện commons-beanutils-1.9.3, commons-logging-1.2 và add vào project.

Bước 3: add file DAM\_lib.jar vào project.

Sau 3 bước đã hoàn thành việc cài đặt cấu hình.

# Hướng dẫn sử dụng:

Bước 1: Tạo file config.xml trong project định nghĩa như sau:

<?xml version="1.0" encoding="UTF-8"?>

<DBConfig id="t1">

<conString id="con1" name="DMA.connection.url">jdbc:sqlserver://localhost:54699;instance=SQLEXPRESS;databaseName=hibernate\_data;integratedSecurity=True</conString>

</DBConfig>

File này sẽ định nghĩa connection String của hệ quản trị cở sở dữ liệu. ở trên là ví dụ connection string của driver jdbc kết nối java vs hệ quản trị SQL Server.

Bước 2: Tạo các lớp mà người dùng cần sử dụng đến và viết các annotation đã định nghĩa sẵn trong thư viện để có thê ánh xạ được bảng dữ liệu dưới cơ sở dữ liệu lên các object. Các thành phần property phải có hàm get set và hàm khởi tạo cho lớp cần dùng.

Ví dụ:

@Table(name = "DEPARTMENT")

**public** **class** DEPARTMENT {

@Primarykey(primarykey = "DEPT\_ID")

@Column(name = "DEPT\_ID")

**private** **int** DEPT\_ID;

@Column(name = "DEPT\_NAME")

**private** String DEPT\_NAME;

@Column(name = "DEPT\_NO")

**private** String DEPT\_NO;

@Column(name = "LOCATION")

**private** String LOCATION;

**public** **int** getDEPT\_ID() {

**return** DEPT\_ID;

}

**public** **void** setDEPT\_ID(**int** dEPT\_ID) {

DEPT\_ID = dEPT\_ID;

}

**public** String getDEPT\_NAME() {

**return** DEPT\_NAME;

}

**public** **void** setDEPT\_NAME(String dEPT\_NAME) {

DEPT\_NAME = dEPT\_NAME;

}

**public** String getDEPT\_NO() {

**return** DEPT\_NO;

}

**public** **void** setDEPT\_NO(String dEPT\_NO) {

DEPT\_NO = dEPT\_NO;

}

**public** String getLOCATION() {

**return** LOCATION;

}

**public** **void** setLOCATION(String lOCATION) {

LOCATION = lOCATION;

}

**public** DEPARTMENT(**int** DEPT\_ID, String DEPT\_NAME, String DEPT\_NO, String LOCATION){

**this**.DEPT\_ID = DEPT\_ID;

**this**.DEPT\_NAME = DEPT\_NAME;

**this**.DEPT\_NO = DEPT\_NO;

**this**.LOCATION = LOCATION;

}

}

Bước 4: Trong hàm main sử dụng SessionFactory để tạo ra các Session làm việc. Phiên làm việc sẽ được mô tả như ví dụ sau:

Các lớp Insert, Delete, Update được kế thừa từ lớp QueryTemplate sẽ cho phép thực thi các lệnh insert, delete, update xuống Database.

**public** **static** **void** main(String[] args) **throws** SQLException, IllegalAccessException, InvocationTargetException, NoSuchMethodException {

// **TODO** Auto-generated method stub

SessionFactory f = SQLServer.*Instance*("D:/workspace2/testDAM/src/config.xml");

Session s = f.createSession();

DEPARTMENT dep = **new** DEPARTMENT(5, "FINANCIAL7", "FCA9", "HOCHIMINH");

Class<?> clazz = dep.getClass();

ArrayList<Class<?>> resList = **new** ArrayList<Class<?>>();

QueryTemplate qt1 = **new** Insert(dep);

qt1.executeQuery(s, clazz, resList);

QueryTemplate qt2 = **new** Delete(dep);

qt2.executeQuery(s, clazz, resList);

QueryTemplate qt3 = **new** Update(dep);

qt3.executeQuery(s, clazz, resList);

}